<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!--

JBoss, Home of Professional Open Source

Copyright 2013, Red Hat, Inc. and/or its affiliates, and individual

contributors by the @authors tag. See the copyright.txt in the

distribution for a full listing of individual contributors.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0* [*http://maven.apache.org/maven-v4\_0\_0.xsd*](http://maven.apache.org/maven-v4_0_0.xsd)*"*>

<modelVersion>4.0.0</modelVersion>

<groupId>org.jboss.tools.examples</groupId>

<artifactId>jboss-javaee-webapp</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>WildFly Quickstarts: jboss-javaee-webapp</name>

<description>A starter Java EE 7 webapp project for use on JBoss WildFly / WildFly, generated from the jboss-javaee6-webapp archetype</description>

<url>http://wildfly.org</url>

<licenses>

<license>

<name>Apache License, Version 2.0</name>

<distribution>repo</distribution>

<url>http://www.apache.org/licenses/LICENSE-2.0.html</url>

</license>

</licenses>

<properties>

<!-- Explicitly declaring the source encoding eliminates the following

message: -->

<!-- [WARNING] Using platform encoding (UTF-8 actually) to copy filtered

resources, i.e. build is platform dependent! -->

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<!-- JBoss dependency versions -->

<version.wildfly.maven.plugin>1.0.2.Final</version.wildfly.maven.plugin>

<!-- Define the version of the JBoss BOMs we want to import to specify

tested stacks. -->

<version.jboss.bom>8.2.1.Final</version.jboss.bom>

<!-- other plugin versions -->

<version.compiler.plugin>3.1</version.compiler.plugin>

<version.surefire.plugin>2.16</version.surefire.plugin>

<version.war.plugin>2.5</version.war.plugin>

<!-- maven-compiler-plugin -->

<maven.compiler.target>1.7</maven.compiler.target>

<maven.compiler.source>1.7</maven.compiler.source>

</properties>

<dependencyManagement>

<dependencies>

<!-- JBoss distributes a complete set of Java EE 7 APIs including a Bill

of Materials (BOM). A BOM specifies the versions of a "stack" (or a collection)

of artifacts. We use this here so that we always get the correct versions

of artifacts. Here we use the jboss-javaee-7.0-with-tools stack (you can

read this as the JBoss stack of the Java EE 7 APIs, with some extras tools

for your project, such as Arquillian for testing) and the jboss-javaee-7.0-with-hibernate

stack you can read this as the JBoss stack of the Java EE 7 APIs, with extras

from the Hibernate family of projects) -->

<dependency>

<groupId>org.wildfly.bom</groupId>

<artifactId>jboss-javaee-7.0-with-tools</artifactId>

<version>${version.jboss.bom}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

<dependency>

<groupId>org.wildfly.bom</groupId>

<artifactId>jboss-javaee-7.0-with-hibernate</artifactId>

<version>${version.jboss.bom}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<dependencies>

<!-- First declare the APIs we depend on and need for compilation. All

of them are provided by JBoss WildFly -->

<!-- Import the CDI API, we use provided scope as the API is included in

JBoss WildFly -->

<dependency>

<groupId>javax.enterprise</groupId>

<artifactId>cdi-api</artifactId>

<scope>provided</scope>

</dependency>

<!-- Import the Common Annotations API (JSR-250), we use provided scope

as the API is included in JBoss WildFly -->

<dependency>

<groupId>org.jboss.spec.javax.annotation</groupId>

<artifactId>jboss-annotations-api\_1.2\_spec</artifactId>

<scope>provided</scope>

</dependency>

<!-- Import the JAX-RS API, we use provided scope as the API is included

in JBoss WildFly -->

<dependency>

<groupId>org.jboss.resteasy</groupId>

<artifactId>jaxrs-api</artifactId>

<scope>provided</scope>

</dependency>

<!-- Import the JPA API, we use provided scope as the API is included in

JBoss WildFly -->

<dependency>

<groupId>org.hibernate.javax.persistence</groupId>

<artifactId>hibernate-jpa-2.1-api</artifactId>

<scope>provided</scope>

</dependency>

<!-- Import the EJB API, we use provided scope as the API is included in

JBoss WildFly -->

<dependency>

<groupId>org.jboss.spec.javax.ejb</groupId>

<artifactId>jboss-ejb-api\_3.2\_spec</artifactId>

<scope>provided</scope>

</dependency>

<!-- JSR-303 (Bean Validation) Implementation -->

<!-- Provides portable constraints such as @Email -->

<!-- Hibernate Validator is shipped in JBoss WildFly -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator</artifactId>

<scope>provided</scope>

<exclusions>

<exclusion>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

</exclusion>

</exclusions>

</dependency>

<!-- Import the JSF API, we use provided scope as the API is included in

JBoss WildFly -->

<dependency>

<groupId>org.jboss.spec.javax.faces</groupId>

<artifactId>jboss-jsf-api\_2.2\_spec</artifactId>

<scope>provided</scope>

</dependency>

<!-- Now we declare any tools needed -->

<!-- Annotation processor to generate the JPA 2.0 metamodel classes for

typesafe criteria queries -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-jpamodelgen</artifactId>

<scope>provided</scope>

</dependency>

<!-- Annotation processor that raising compilation errors whenever constraint

annotations are incorrectly used. -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator-annotation-processor</artifactId>

<scope>provided</scope>

</dependency>

<!-- Needed for running tests (you may also use TestNG) -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<scope>test</scope>

</dependency>

<!-- Optional, but highly recommended -->

<!-- Arquillian allows you to test enterprise code such as EJBs and Transactional(JTA)

JPA from JUnit/TestNG -->

<dependency>

<groupId>org.jboss.arquillian.junit</groupId>

<artifactId>arquillian-junit-container</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.jboss.arquillian.protocol</groupId>

<artifactId>arquillian-protocol-servlet</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<!-- Maven will append the version to the finalName (which is the name

given to the generated war, and hence the context root) -->

<finalName>${project.artifactId}</finalName>

<plugins>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>${version.war.plugin}</version>

<configuration>

<!-- Java EE 7 doesn't require web.xml, Maven needs to catch up! -->

<failOnMissingWebXml>false</failOnMissingWebXml>

</configuration>

</plugin>

<!-- The WildFly plugin deploys your war to a local WildFly container -->

<!-- To use, run: mvn package wildfly:deploy -->

<plugin>

<groupId>org.wildfly.plugins</groupId>

<artifactId>wildfly-maven-plugin</artifactId>

<version>${version.wildfly.maven.plugin}</version>

</plugin>

</plugins>

</build>

<profiles>

<profile>

<!-- The default profile skips all tests, though you can tune it to run

just unit tests based on a custom pattern -->

<!-- Seperate profiles are provided for running all tests, including Arquillian

tests that execute in the specified container -->

<id>default</id>

<activation>

<activeByDefault>true</activeByDefault>

</activation>

<build>

<plugins>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>${version.surefire.plugin}</version>

<configuration>

<skip>true</skip>

</configuration>

</plugin>

</plugins>

</build>

</profile>

<profile>

<!-- An optional Arquillian testing profile that executes tests

in your WildFly instance -->

<!-- This profile will start a new WildFly instance, and execute the

test, shutting it down when done -->

<!-- Run with: mvn clean test -Parq-wildfly-managed -->

<id>arq-wildfly-managed</id>

<dependencies>

<dependency>

<groupId>org.wildfly</groupId>

<artifactId>wildfly-arquillian-container-managed</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

</profile>

<profile>

<!-- An optional Arquillian testing profile that executes tests

in a remote WildFly instance -->

<!-- Run with: mvn clean test -Parq-wildfly-remote -->

<id>arq-wildfly-remote</id>

<dependencies>

<dependency>

<groupId>org.wildfly</groupId>

<artifactId>wildfly-arquillian-container-remote</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

</profile>

<profile>

<!-- When built in OpenShift the 'openshift' profile will be used when

invoking mvn. -->

<!-- Use this profile for any OpenShift specific customization your app

will need. -->

<!-- By default that is to put the resulting archive into the 'deployments'

folder. -->

<!-- <http://maven.apache.org/guides/mini/guide-building-for-different-environments.html> -->

<id>openshift</id>

<build>

<plugins>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>${version.war.plugin}</version>

<configuration>

<outputDirectory>deployments</outputDirectory>

<warName>ROOT</warName>

</configuration>

</plugin>

</plugins>

</build>

</profile>

</profiles>

</project>

pom