woapplication-archetype

Introduction

The woapplication-archetype is a template to create WebObjects applications. If you start a project using this archetype, all the basic WebObjects files are generated following the standard directory layout.

This archetype requires the most recent version of maven-archetype-plugin 2.0-alpha-X or later. See maven-archetype-woapplication if you want to use an older version of maven-archetype-plugin.

Usage

Warning

At the time of writing, the Archetype plug-in was in a very early alpha version. This plug-in uses catalogs to know about specific archetypes. You can download the archetype catalog with information about the woapplication-archetype here. You have to save this file into your local repository (.m2) or merge the contents if a file with the same name already exists.
Currently (February 2009) this creates wonder supported projects that refer to a groupid of wonder.common, this should be wonder.core.

This archetype allows you to generate a basic project for a WebObjects application. To use this archetype execute the following command:

```
mvn archetype:generate -DarchetypeCatalog=local
```

First timers

Ensure you have the latest archetype-catalog.xml file to ensure you're referencing the correct repository and artifacts

Choose the archetype:

```
1: local -> woapplication-archetype (WebObjects Application Archetype)
```

Define the project properties:

- **groupId** - the groupId for the new project pom.xml.
- **artifactId** - the artifactId for the new project.
- **version** - an initial version for your project.
- **package** - the package where WebObjects basic classes will be placed.
- **WebObjectsVersion** - the WebObjects version used in the project. The default value for this property is 5.4.1.
- **WonderSupport** - type 'yes' if you want the Wonder framework support in your classes. The default value for this property is 'no'.
- **TrueWarSupport** - type 'yes' if you want to build your project as a true WAR instead of WOA. The default value for this property is 'no'.

See Building the wonder source code with maven (http://wiki.objectstyle.org/confluence/display/WOL/Building+the+wonder+source+code+with+maven) for selecting the correct versions of Wonder (for WebObjects 5.3.x or 5.4.x).

Standard Directory Layout for WebObjects Applications

"Having a common directory layout would allow for users familiar with one Maven project to immediately feel at home in another Maven project. The advantages are analogous to adopting a site-wide look-and-feel." (http://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html)

The woapplication-archetype provides some additional directories to conform to WebObjects development in addition to the Maven Standard Directory Layout:

```
my-app
|-- pom.xml
`-- src
    `-- main
        |-- java
        |   `-- my
        |       `-- group
        |           |-- app
        |           |    |-- Application.java
        |           |    |-- DirectAction.java
        |           |    `-- Session.java
        |           |
        |           `-- components
        |                `-- Main.java
        |
        |-- resources
        |   |-- Info.plist
        |   |-- MyModel.eomodeld
        |   `-- Properties
        |
        |-- components
        |   |-- Main.api
        |   `-- Main.wo
        |       |-- Main.html
        |       |-- Main.wod
        |       `-- Main.woo
        |
        |-- webapp
        |   `-- WEB-INF
        |       |-- web.xml
        |       `-- LICENSE
        |
        `-- webserver-resources
            |-- images
            |   `-- sample.jpg
            |
            |-- css
            |   `-- sample.css
            |
            `-- flash.swf
```

Warning

All resources and webserver-resources structured inside sub folders (i.e. webserver-resources/css/sample.css) will be packaged with the same structure (i.e. WebServerResources/css/sample.css). maven-wolifecycle-plugin has an option (flattenResources) to automatically flatten these resources.

If you want more control over this configuration, you can configure the resources in your pom.xml like this:

```
<build>
...
   <resources>
      <resource>
         <targetPath>WebServerResources</targetPath>
         <directory>
            ${basedir}/src/main/webserver-resources
         </directory>
         <includes>
            <include>*.swf</include>
         </includes>
      </resource>
      <resource>
         <targetPath>WebServerResources</targetPath>
         <directory>
            ${basedir}/src/main/webserver-resources/images
         </directory>
         <includes>
            <include>*.jpg</include>
         </includes>
      </resource>
      <resource>
         <targetPath>WebServerResources</targetPath>
         <directory>
            ${basedir}/src/main/webserver-resources/css
         </directory>
         <includes>
            <include>*.css</include>
         </includes>
      </resource>
   </resources>
...
</build>
```