

Split Install Deployment

Introduction

Below is outlined "simple embedding" concepts, however **full embedding and split-installing** are really recommended. This provides fully versioned self-contained bundles of both the application deployment bundle and the webserver deployment bundle. Read the docs on that technique for more **advantages**. This technique fully explained along with a ready-to-use ant build script is available at:

[Alternative Ant Build Script for Fully Embedded and Split Install Bundles](#)

The Simple Way

One simple change to your build.xml: Simply add the `wsDestDir` property to the `woapplication` (or `woframework` in the case of a framework project) task in your project's build.xml as shown below. As stated by the [documentation](#), presence of that parameter triggers a split install and thus creates an additional `woa` bundle for the webserver resources.

woapplication opening tag with wsDestDir property

```
<woapplication name="${project.name}" stdFrameworks="false"
               destDir="${dest.dir}"
               customInfoPListContent="${customInfoPListContent}"
               principalClass="${principalClass}"
               webXML="${webXML}"
               webXML_CustomContent="${webXML_CustomContent}"
               wsDestDir="/Library/WebServer/Documents">
```

woframework opening tag with wsDestDir property

```
<woframework name="${framework.name}"
              destDir="${dest.dir}"
              customInfoPListContent="${customInfoPListContent}"
              principalClass="${principalClass}"
              eoAdaptorClassName="${eoAdaptorClassName}"
              wsDestDir="/Library/WebServer/Documents">
```

A Better Way

So that both Install works and ant build into project dist (right-click build.xml -> Run As/Ant Build) provides a split install, make the following 4 1-line changes to your Application build.xml file as shown in snippet below:

```
setProps target : add wsinstall.dir property
init.install target : add wsdest.dir property
init.build target : add wsdest.dir property
woapplication task : add wsDestDir property
```

```

<!-- property determination -->
<target name="setProps">
  <property file="${user.home}${file.separator}build.properties"/>
  <property file="build.properties"/>
  <property file="${user.home}${file.separator}Library${file.separator}wobuild.properties"/>
  <condition property="wo.properties.check.failed">
    <not>
      <and>
        <isset property="wo.wosystemroot"/>
        <isset property="wo.wolocalroot"/>
      </and>
    </not>
  </condition>
  <fail message="Could not find ${user.home}${file.separator}Library${file.separator}wobuild.properties."
if="wo.properties.check.failed"/>
  <property name="install.dir" value="${wo.wolocalroot}/Library/WebObjects/Applications"/>
  <property name="wsinstall.dir" value="${wo.wolocalroot}/Library/WebServer/Documents"/>
</target>

<!-- basic initializations -->
<target name="init.install">
  <tstamp/>
  <property name="dest.dir" value="${install.dir}"/>
  <property name="wsdest.dir" value="${wsinstall.dir}"/>
</target>

<target name="init.build">
  <tstamp/>
  <property name="dest.dir" value="dist"/>
  <property name="wsdest.dir" value="dist/wsdocroot"/>
</target>

<!-- woproject tasks -->
<target name="build.woapp">

  <taskdef name="woapplication" classname="org.objectstyle.woproject.ant.WOApplication">
  </taskdef>

  <!-- add webXML="true" to generate a web.xml file -->
  <woapplication name="${project.name}" stdFrameworks="false"
    destDir="${dest.dir}"
    customInfoPListContent="${customInfoPListContent}"
    principalClass="${principalClass}"
    webXML="${webXML}"
    webXML_CustomContent="${webXML_CustomContent}"
    wsDestDir="${wsdest.dir}">

```