

Configure

Table of contents

| | |
|---|---|
| 1 Step #1: Edit defaults.xml..... | 2 |
| 2 Step #2: Configure library objects..... | 4 |
| 3 Step #3: Upload job definition..... | 6 |
| 4 Step #4: Upload packages to the repository..... | 6 |

Configure

```
<!-- This file can be supplied to the ProjectBuilder "generate-objects"
command to load -->
<!-- sample objects into a project that contains the "content" library's
type model. -->
<!--
-->
<!-- -----
----- -->

<defaults>
  <default>
    <!-- The default node is the framework node of the Antdepo
client invoking the -->
    <!-- generate-objects command. (For the default
installation this will be -->
    <!-- "localhost").
-->
    <node>${framework.node}</node>
  </default>

  <!-- The default location of the installation directory for the
Duke's bank -->
  <!-- application source files and packages:
-->
  <dukesbankroot>${user.home}/dukesbank</dukesbankroot>

  <!-- The out-of-the box Duke's Bank demonstration can be configured
to run on -->
  <!-- from one to six systems. One system must be designated the
ControlTier -->
  <!-- server. This is the system where Workbench and Jobcenter run.
Another -->
  <!-- system is designated as the build box where CruiseControl and
the Ant -->
  <!-- based build are run.
-->
  <!--
-->
  <!-- One or two boxes are assigned to the development environment.
Builds are -->
  <!-- run here, and the development JBoss instance and "source"
HSQLDB database -->
  <!-- instance are deployed here. A fifth and possibly sixth system
host the -->
  <!-- staging environment which includes both JBoss and HSQLDB
instances. -->
  <!--
-->
  <!-- Note that all these systems can be the same box in which case
separate -->
  <!-- JBoss and HSQLDB server instances are started on separate
ports. -->

  <server>
```

```

        <node>${defaults.default.node}</node>
    </server>
    <development>
        <buildserver>
            <!-- The default copy of the JBoss 4.0 modified
Duke's Bank source code is kept under Subversion on Sourceforge: -->
<scmconnection>https://moduleforge.svn.sourceforge.net/svnroot/moduleforge/elements/bra
            <node>${defaults.default.node}</node>
        </buildserver>
        <applicationserver>
            <node>${defaults.default.node}</node>
        </applicationserver>
        <databaseserver>
            <node>${defaults.default.node}</node>
        </databaseserver>
    </development>
    <staging>
        <applicationserver>
            <node>${defaults.default.node}</node>
        </applicationserver>
        <databaseserver>
            <node>${defaults.default.node}</node>
        </databaseserver>
    </staging>
</defaults>

```

You are at liberty to change any of the six box names to systems that make sense in your environment, thereby ending up with the sample Duke's Bank development and staging environments deployed to as many as six and as few as a single system.

Note:

A copy of the defaults.xml template file can be found in the WebDAV under your project (substituting a host name for "localhost" if the ControlTier server is deployed remotely):
 "http://localhost:8080/webdav/project/modules/ElementsProjectBuilder/templates/defaults.xml" ... where *project* is "DukesBank" if you are following the demonstration setup.

2. Step #2: Configure library objects

Register and install an ElementsProjectBuilder object:

```

ad -p project -m Deployment -c Register -- \
    -name name -type ElementsProjectBuilder \
    -basedir $CTIER_ROOT/src/project -installroot
$CTIER_ROOT/target/project \
    -install

```

... or, specifically for the Duke's Bank demonstration:

Configure

```
$ ad -p DukesBank -m Deployment -c Register -- -name elements -type
ElementsProjectBuilder -basedir $CTIER_ROOT/src/elements -installroot
$CTIER_ROOT/target/elements -install
Checking for existing object, (ElementsProjectBuilder) elements, in
project, 'DukesBank'...
Registered new object.
.
.
.
[command.timer.Deployment.Register: 3.096 sec]

C:\>ad -p DukesBank -m Deployment -c Register -- -name elements -type
ProjectBuilder -basedir %CTIER_ROOT%\src\elements -installroot
%CTIER_ROOT%\target\elements -install
.
.
.
[command.timer.Deployment.Register: 2.359 sec]
```

Copy the defaults.xml you created in [Step #1](#) to `$CTIER_ROOT/src/project/defaults.xml` and run the generate-objects command:

```
ad -p project -t ElementsProjectBuilder -o name -c generate-objects -- \
    -name aName \
    -defaults $CTIER_ROOT/src/project/defaults.xml -upload
```

... or, specifically for the Duke's Bank demonstration (using different defaults XML files for Linux and Windows):

```
$ ad -p DukesBank -t ElementsProjectBuilder -o elements -c generate-objects
-- -defaults
$ANTDEPO_BASE/depots/DukesBank/lib/ant/modules/ElementsProjectBuilder/templates/default
-upload
.
.
.
[command.timer.generate-objects: 0.156 sec]
generate-objects completed. execution time: 0.156 sec.
```

After this command successfully completes, a new set of objects will be loaded into the ControlTier repository. You can view them via ElementsProjectBuilder's find-objects command:

```
ad -p project -t ElementsProjectBuilder -o name -c find-objects -- \
    -name aName
```

Before you can run any commands, it is necessary to deploy the objects. This is done via the AntDepo command, `depot-setup`. Run the following installation command on all boxes in your configuration:

```
depot-setup -p project -a install
```

... or, if you're following the Duke's Bank demonstration setup instructions, more specifically

(for a single box example):

```
$ depot-setup -p DukesBank -a install
"Install" command running for object: (Site) developmentCI
"Install" command running for object: (HsqldbRdb) staging
"Install" command running for object: (Updater) development
"Install" command running for object: (HsqldbRdb) development
"Install" command running for object: (Site) developmentApplicationServer
"Install" command running for object: (ElementsProjectBuilder) elements
"Install" command running for object: (Site) stagingDatabaseServer
"Install" command running for object: (CruiseControl) development
"Install" command running for object: (AntBuilder) development
"Install" command running for object: (Site) stagingApplicationServer
"Install" command running for object: (Site) developmentDatabaseServer
"Install" command running for object: (HsqldbRdbExportBuilder) development
"Install" command running for object: (Updater) staging
"Install" command running for object: (JBossServer) development
"Install" command running for object: (JBossServer) staging
```

3. Step #3: Upload job definition

The `generate-objects` command run in [Step #2](#) will have produced a `job.xml` file with a filename `aName-job.xml`. This file can be used to define a new job in the JobCenter application.

1. Login to JobCenter (e.g, go to URL: <http://localhost:9090/jobcenter/menu/index>)
2. Press the "Create a new Job..." button
3. Press the "Upload job.xml" button
4. Locate and select the file, `aName-job.xml`, produced by `generate-objects` in the file chooser
5. Press "Save" button

The new job will be listed on the home page of JobCenter.

4. Step #4: Upload packages to the repository

The solution library also manages all the platform (3rd party) software packages needed to establish environments. The only assumptions are that you have a compatible OS image at your disposal, a user account with the ControlTier client installed and available, and sufficient disk space to deploy the platform and application software.

Note:

The Duke's Bank demonstration assumes you have a system with Java and Subversion pre-installed.

The general method of adding 3rd party packages to the ControlTier repository is by

uploading them using Workbench:

If your intention is to run the Duke's Bank demonstration and you have generated and configured the example library objects mentioned above, then upload the following packages into the ControlTier repository via Workbench (i.e. select package objects and upload one at a time):

- The Zip archive of the binary distribution of Ant 1.7.0 (<http://ant.apache.org/bindownload.cgi>).
- The Zip archive of the binary distribution of CruiseControl 2.7.1 (http://sourceforge.net/project/showfiles.php?group_id=23523&package_id=16338).
- The Zip version of the JBoss 4.0.3SP1 subscription or community edition (<http://labs.jboss.com/jbossas/downloads/>).
- The Zip archive of HSQLDB 1.8.0.9 (http://sourceforge.net/project/showfiles.php?group_id=23316&package_id=16653).
- The initial version of the Duke's Bank data package from Moduleforge (<http://moduleforge.svn.sourceforge.net/viewvc/moduleforge/elements/branches/2.0/demo/DukesBank/d>

[Next: Run #](#)