

Installing AntDepo

1. Installing AntDepo

The binary distribution of AntDepo consists of the following directory layout:

```
antdepo
+--- bin      // contains launcher scripts
+--- classes // contains bootstrapping classes
+--- etc      // framework configuration files
+--- lib      // contains core framework modules
+--- pkgs     // contains Ant, AntDepo jars plus necessary dependencies
```

To install Ant, choose a directory and copy the distribution file there. This directory will be known as `ANTDEPO_HOME`.

Example: Installing from a zip archive

```
export ANTDEPO_HOME=$HOME/antdepo-1.2.6
mkdir $ANTDEPO_HOME
cp /path/to/antdepo-1.2.6.zip $ANTDEPO_HOME/
cd $ANTDEPO_HOME && unzip antdepo-1.2.6.zip
chmod +x $ANTDEPO_HOME/bin/* # necessary if installing from zip archive
```

2. Setup

Before you can run the AntDepo setup you will need to:

- Set the `JAVA_HOME` environment variable to the directory where you installed the Java runtime. (Note: 1.4.2 is the "officially" supported version.)
- Set the `ANTDEPO_HOME` environment variable to the directory where you installed AntDepo.
- Add the `$ANTDEPO_HOME/bin` directory to your path
- Set the `ANTDEPO_BASE` environment variable to the directory where you decided the the AntDepo repository will be kept.

Run the `ad-setup` command

```
$ANTDEPO_HOME/bin/ad-setup -n hostname
```

Example

```
export ANTDEPO_BASE=$HOME/antdepo_base
mkdir $ANTDEPO_BASE
$ANTDEPO_HOME/bin/ad-setup -n `hostname`
```

3. Extensions

The AntDepo framework can be extended via [AntDepo's extension model](#). Extensions can include new Ant typedefs, jars, bins and modules. Extensions are installed via the ext-setup command. Before you can install an extension you will need:

- An AntDepo framework installation
- A framework instance setup with ad-setup

Run the ext-setup command

```
$ANTDEPO_HOME/bin/ext-setup -f extension-name.jar
```

The ext-setup unpacks the extension and then runs the ad-setup process again. You can also specify various paramaters to the ext-setup command as its usage shows below:

```
usage: ext-setup [options]
options:
  -S,--nosetup           don't re-run setup
  -D <property=value>   property=value
  -f,--file              file to install
  -h,--help             print this message
  -o,--overwrite        overwrite framework with files from extension
Examples:
ext-setup -f extension-name.jar
```

4. Depot Setup

AntDepo allows you to organize modules and objects into project "depots". Depots allow you to partition your process into separate workspaces. You might choose to create a depot for managing a process in a new environment or might choose to designate a depot for managing a related set of services.

The depot-setup command provides a set of administrative actions for creating, updating and removing depots.

The depot-setup command takes a number of options as shown below:

```
usage: depot-setup [options]
options:
  -n,--name              project depot name. deprecated. use -p
  -d,--name              project depot name. deprecated. use -p
  -D,--deploy           run deploy action.
  -L,--depotmoduledir   depot module directory
  -M,--packagedmoduledir packaged module directory
  -S,--strict           strictly use the registration info from the depot
                       deployments.properties
```

Installing AntDepo

```
-a,--action          action to run {create | deploy | remove | undeploy}
-b,--buildfile       ant build file to run
-f,--file            deployments.properties file
-h,--help           print this message
-p,--name           project depot name
-v,--verbose        verbose messages
Examples:
depot-setup -p depot          # create depot
depot-setup -p depot -a create # same as above
depot-setup -p depot --action deploy # update deployments in depot
depot-setup -p depot --action remove # archives and removes the depot
```

5. Central Repository (aka WebDAV)

For users wishing to maintain their modules in a central repository, they can be located on a WebDAV repository, using a standard structure.

Repository Structure

```
webdav
+---- pkgs // contains release artifacts for managed project depots
+---- ProjectA // contains release artifacts for depot, projectA
+---- pType // contains release artifacts of package type, pType
+---- pExt // contains pType packages with file extension, pExt
+---- aFile.pExt // a pType release artifact
+---- ProjectA // contains modules and configuration for depot, ProjectA
+---- publish
+---- modules // contains packaged module jars for projectA
+---- etc // contains ProjectA specific framework configuration
+---- modules // contains ProjectA specific module sources
+---- ProjectB // contains release artifacts for ProjectB
```

Note:

Users interested in employing a WebDAV based repository but do not wish to manually maintain one, can consider the [ControlTier Server](#).