

Writing a Simple Module

1. Overview

This document describes how to create a trivial AntDepo module from scratch without the use of any special tools. Creating the module by hand, shows off the basic requirements to develop a module. AntDepo modules use a simple, standard file structure with a couple of required configuration property files.

Modules have the following directory layout:

```
module_name
|
|--- module.properties // file containing module metadata
|--- commands.properties // file containing command metadata
+--- bin // optional binaries, shell scripts, etc.
|
+--- commands // contains command handlers
|
+--- lib // optional module resource files
```

2. 1. Create the Module Structure

Create a directory for the module in the module library:

```
mkdir $ANTDEPO_BASE/lib/ant/modules/HelloModule
mkdir $ANTDEPO_BASE/lib/ant/modules/HelloModule/commands
cd $ANTDEPO_BASE/lib/ant/modules/HelloModule
```

Create the `module.properties` file. The `module.properties` file is a standard configuration property file describing the module and its contents. Name and describe the module and define a command named `Hello`.

```
module.name=HelloModule
module.description=My first module
module.commands=Hello
```

3. 2. Create a Command

With the module structure built, the next step is to create an implementation for the `Hello` command by supplying a command handler (ie., an Ant buildfile). Create a command handler file named `Hello.xml` and save it to `$ANTDEPO_BASE/lib/ant/modules/HelloModule/commands`.

Write Hello.xml

```
<project name="Hello" default="execute">
  <property file="${module.dir}/module.properties"/>
  <target name="execute">
    <echo>Hi there</echo>
  </target>
</project>
```

AntDepo command handlers are Ant buildfiles that follow a few conventions. The primary requirement is that the handler has a target named `execute` and that target should be declared default for the project.

4. 3. Run the command

Run the command by specifying module name and command name.

```
ad -m HelloModule -c Hello
Hi there
```

5. Creating a Shell command type

This section describes how to define a command that uses a `shell` command type. A shell command type uses configuration data to define the script code to run. Declaring the script code in a property file allows you to define a generic handler template.

1. Add the command to the `module.properties` file

```
module.name=HelloModule
module.description=My first module
module.commands=Hello, Echo
```

2. Define the command in the `commands.properties` file

The next step is to describe the shell command in the `commands.properties` file in the module base directory. Define the four properties as shown below.

```
command.Echo.command-type=shell
command.Echo.execution-string=bash
command.Echo.argument-string=echo Hi there
command.Echo.doc=Says hello
```

3. Write Echo.xml

Create the handler for the Echo command using the [ant-contrib](#) shellsript task as shown below. Note the `commands.properties` file is now read.

```
<project name="Echo" default="execute">
  <property file="${module.dir}/module.properties"/>
```

Writing a Simple Module

```
<property file="${module.dir}/commands.properties"/>
<target name="execute">
  <shellscript shell="${command.Echo.execution-string}">
    ${command.Echo.argument-string}
  </shellscript>
</target>
</project>
```

4. Run the command

```
ad -m HelloModule -c Echo
Hi there
```