

Getting Around

Table of contents

1 Overview.....	2
2 Tools.....	2
2.1 ControlTier Server (Repository and Workbench).....	2
2.2 ControlTier Agent (Execute Commands).....	2

1. Overview

This section describes the environments and tools used to carry out the routine procedures using the release management system.

2. Tools

The release management system is built using the open source [ControlTier](#) platform.

A ControlTier project named, `headlines`, should be used to manage the automation and release artifacts for the Headlines-News Services application.

2.1. ControlTier Server (Repository and Workbench)

The ControlTier server maintains a repository of release artifacts and management automation modules. The web-based application, [Workbench](#), provides a graphical interface to defining the deployment and control model for the Headlines applications.

At this time, a single project named, `headlines`, defines the repository for the Headlines environments and release processes.

2.2. ControlTier Agent (Execute Commands)

The release manager uses the ControlTier agent to perform the release process. The ControlTier agent uses the [AntDepo](#) framework to execute these procedures as AntDepo commands.

After logging into a machine, the `ad` command can be run without arguments to see what projects and objects are available to manage.

The example below shows successive invocations of the `ad` commands showing how to use `ad` to query the local repository.

```
$ ad
#Available projects :
headlines

$ ad -p headlines
#Available types in project:
AntBuilder
Apache
JakartaTomcat
MysqlRdb
MysqlSchema
```

```
Node
ProjectBuilder
Site
WarUpdater

$ ad -p headlines -t Site
#Available instances:
headlines

$ ad -p headlines -t Site -o headlines
#Available commands
Start
Status
Stop
...
```

This last invocation of `ad` shows the commands available for the specified object.

2.2.1. Running Commands

Commands can be executed by specifying the `-c` command argument and providing a command name as a parameter.

```
$ ad -p headlines -t Apache -o headlines -c Status
running assertServiceIsUp command
Apache web service is up.
[command.timer.Service.Status: 3.581 sec]
```

The `Status` command ran and completed successfully.

Note:

Successful commands return an exit code of 0. For example,
[\$? -eq 0] && echo success => success

The following example shows a `Stop` command.

```
$ ad -p headlines -t Apache -o headlines -c Stop
running command: assertServiceIsDown
Running handler command: stopService
Apache service stopped.
[command.timer.Service.Stop: 4.758 sec]
```

This output shows the `Stop` command is actually a workflow type command which runs the `assertServiceIsDown` command. The `assertServiceIsDown` command will return true if the apache instance is not running. If it fails, that means the apache instance is running, and the workflow error handler runs the `stopService` command.

Finally, the `Start` command is used to bring up the Apache instance:

```
$ ad -p headlines -t Apache -o headlines -c Start
```

```
running command: assertServiceIsUp
Running handler command: startService
Apache service started.
[command.timer.Service.Start: 4.446 sec]
```

Similar to the Stop command, the Start command is also a command workflow. It first runs the `upService` command and if that fails, the workflow error handler runs the `startService` command.

These examples have shown commands which complete successfully. To contrast those with one that shows the output of a failure, the next example demonstrates a failed command.

The last command started apache successfully, that means if the `assertServiceIsDown` command is run now, it should fail:

```
$ ad -p headlines -t Apache -o headlines -c assertServiceIsDown
Command failed.
ctier/antdepo/depots/headlines/lib/ant/modules/Apache/commands/assertServiceIsDown.xml:
Apache web service is Up.

error: Apache web service is Up.
```

The output of the command shows that the `assertServiceIsDown` command finds the Apache server is running and therefore fails.

Note:

Commands that fail, return a non 0 exit status. For example,
`[ $? -eq 0 ] || echo failed=> failed`

Refer to the base-type documentation which discusses the routine procedures of [building](#), [updating](#), and [operating](#) the Headlines application sites.