Installing the Cisco Prime Network Registrar SDK

This section documents how to install the Cisco Prime Network Registrar SDK on the Linux and Windows platforms. Before installing the SDK, ensure that you have Java Runtime Environment (JRE) 1.7 or later, or the equivalent Java Development Kit (JDK), installed on your system. The Cisco Prime Network Registrar SDK is a separate product and is sold separately

This appendix contains the following sections:

Installing on Linux

To install the Cisco Prime Network Registrar SDK on a Linux platform:

Step 1   Extract the contents of the distribution .tar file.

a)   Create the SDK directory:

```
% mkdir /cnr-sdk
```

b)   Change to the directory that you just created and extract the .tar file contents:

```
 % cd /cnr-sdk

% tar xvf sdk_tar_file_location/cnrsdk.tar
```

Step 2   Export your LD_LIBRARY_PATH and CLASSPATH environment variable:

```
 % export LD_LIBRARY_PATH=/cnr-sdk/lib

% export CLASSPATH=/cnr-sdk/classes/cnrsdk.jar:.
```

Installing on Windows

To install the Cisco Prime Network Registrar SDK on a Windows platform:

Step 1   Extract the contents of the distribution .tar file.

a)   Create the SDK directory:

```
> md c:\cnr-sdk
```

b)   Change to the directory that you just created and extract the .tar file contents:

```
 > c:

> cd \cnr-sdk

> tar xvf sdk_tar_file_location\cnrsdk.tar
```

You may optionally use Winzip to extract cnrsdk.tar to the C:\cnr-sdk directory.

Step 2   Set your PATH and CLASSPATH variables:

```
 > set PATH=%PATH%;c:\cnr-sdk\lib

> set CLASSPATH=c:\cnr-sdk\classes\cnrsdk.jar;.
```

Testing Your Installation

On Linux, the following test program verifies that you have set your PATH or LD_LIBRARY_PATH correctly:

```
% java -jar /cnr-sdk/classes/cnrsdk.jar
```

On Windows, the following test program verifies that you have set your CLASSPATH correctly:

```
> java -jar c:\cnr-sdk\classes\cnrsdk.jar
```

Compatibility Considerations

For Java SDK client code developed with an earlier version of the SDK, you can simply recompile most code with the latest JAR file to connect to an upgraded server.

But in cases where the client code for versions before 7.1 directly manipulates reservation lists in scopes or prefixes, changes are required. These changes are required because the embedded reservation lists in both scopes and prefixes are no longer used. Beginning with version 7.1, individual reservations are stored separately and reference the parent scope or prefix by name.

The new design provides the following benefits:

• Reservation edits (add/modify/delete) do not require a scope or prefix edit.

• Reservations can be indexed directly to allow quick search and retrieval.

- Edits to scopes or prefixes with a large number of reservations no longer result in large scope or prefix change entry logs.

No changes are required for client code that adds or removes reservations using the addReservation or removeReservation methods. However, these methods are now deprecated because the edit functionality is replaced and extended by the general addObject, modifyObject, removeObject, addObjectList, modifyObjectList, and removeObjectList methods.