Managing Annotations and LOVs

This chapter contains the following sections:

Annotations

Annotations are one of the most crucial parts of Module development. Most of the artifacts are driven by annotations. This makes the development effort all the more easy and convenient.

Annotations are used for persistence, report generation, wizard generation, and tasks.

Persistence Annotations

See Marking a Class for Persistence, for information about the annotations that are used for persistence.

Task Annotation

When a task is included in a Workflow, the user is prompted for certain inputs. The user is prompted for an input when a field of the class representing the task is marked with an annotation. The FormField annotation determines what type of UI input field to show to the user: a text field, or a dropdown list, or a checkbox, etc. For more information, see Tasks.

Lists of Values (LOVs)

Lists represent the drop-down LOVs (Lists of Values) that are displayed to the user to facilitate getting the correct inputs for a task. You can reuse an existing list or create your own list to show in the Task UI.

Cisco UCS Director defines over 50 prebuilt List providers that the modules can readily use to prompt input from the user. For more information, see Appendix A.

For an example that shows how to use one of the list providers, see Defining Your Own List Provider, on page 1, and Tasks.

Defining Your Own List Provider

You can define your own list provider and ask the Platform Runtime to register it with the system.

A list provider class implements the **LOVProviderIf** interface and provides implementation for the single method **getLOVs( )**. See the following example:

```
class MyListProvider implements LOVProviderIf
{

 /**
  * Returns array of FormLOVPair objects.  This array is what is shown
  * in a dropdown list.
  * A FormLOVPair object has a name and a label. While the label is shown
  * to the user, the name will be used for uniqueness
  */
 @Override
 public FormLOVPair[] getLOVs(WizardSession session) {

  // Simple case showing hard-coded list values

  FormLOVPair http = new FormLOVPair("http", "HTTP");
  // http is the name, HTTP is the value
  FormLOVPair https = new FormLOVPair("https", "HTTPs");

  FormLOVPair[] pairs = new FormLOVPair[2];
  pairs[0] = http;
  pairs[1] = https;
  return pairs;
 }
}
```