Open Service Interface Definition

Id

Document Release: 2.0
OSID version 2.0

Summary

The Id Open Service Interface Definition (OSID) supports managing and using Ids.

Service Definition

An examination of an Open Service Interface Definition (OSID) usually begins with the Manager. All Managers[1] provide the way to create the objects that implement the principal interfaces in the Service. Before looking at the OSID in detail, there are several concepts to review:

Ids

The id OSID supports creating, storing, and retrieving unique identifiers (Ids). In many contexts, people, organizations, objects, and processes are designated both a descriptive name and an unambiguous identifier. For example, students have names and student ids; courses have titles and course numbers; books have titles and ISBNs, web pages have names and URLs, etc. The convention is that names do not have to be unique but Ids should be, if not globally, at least within some broad context.

Ids are used in many OSIDs and the definition of Id is located in shared OSID. The id OSID defines the service interface for support of creation, storage, and retrieval of Ids. There can be multiples implementations for the id OSID. Each implementation typically supports Ids from a specific domain. Ids created in one domain do not necessarily have meaning in another domain. See the shared OSID for information about the form of an Id.

org.osid.id.IdManager

The IdManager includes the **createId**() method. This method accepts no arguments and returns a new, unique Id. The implementation of this method is not specified by the OSID but no two Ids created through this method should be equal. The **getId**() method accepts a String representation and reconstitutes the Id with this representation if one has already been created or creates a new Id with this String representation. Note that there is no Service to delete an Id. Once created, they are never removed.

[1] org.osid.OsidManager defines the interface extended by the Managers in each OSID. Here we will be discussing org.osid.id.IdManager. Refer to OsidManager for more information.

OSID Managers are the primary interface through which objects are created, although in some OSIDs objects are created elsewhere. In each case, creation implies that the resulting object is stored and that it can be retrieved. A form of “create” and “get” methods are included in this approach. The implementation of the “create” method will usually create an Id for the new object and the object will have a **getId()** method to return this Id.[2] Unlike descriptions and in some cases display names, Ids are immutable in these objects.

IdManager Method Summary

Id	createId()
Id	getId(String idString)

org.osid.id.IdException

The OSIDs make use of Exceptions as a mechanism for responding to error or unusual conditions. All methods in the Id OSID throw an **IdException**. The Exception contains a message that is a String.

If an implementation uses these messages, consumers of the implementation can easily test and conditionally respond to the Exception. Note that-other kinds of Exception constructors are not used as all do or can devolve to a String. All methods of all interfaces of all OSIDs throw a subclass of org.osid.OsidException. This requires the caller of any implementation method handle the Exception.

If a method performs an operation without incident, an object or primitive may be returned, but in most cases, methods do not return error codes or a success or failure boolean. For example, a method that deletes an object with a particular identifier, would throw an Exception if the identifier were unknown; the method would not return, for example, false.

[2] All OSIDs are intended to standalone and any implementation should be able to be substituted for another. If an OSID uses a particular implementation of IdManager, that Manager will be responsible for persisting any Ids created. If, later, another implementation of IdManager is used, it may not reference the same persistence store. In this case, using the getId() method will create any missing Ids.