Client-Side Slice-to-Java Mapping

In this section, we present the client-side Slice-to-Java mapping. The client-side Slice-to-Java mapping defines how Slice data types are translated to Java types, and how clients invoke operations, pass parameters, and handle errors. Much of the Java mapping is intuitive. For example, Slice sequences map to Java arrays, so there is essentially nothing new you have to learn in order to use Slice sequences in Java.

The Java API to the Ice run time is fully thread-safe. Obviously, you must still synchronize access to data from different threads. For example, if you have two threads sharing a sequence, you cannot safely have one thread insert into the sequence while another thread is iterating over the sequence. However, you only need to concern yourself with concurrent access to your own data — the Ice run time itself is fully thread safe, and none of the Ice API calls require you to acquire or release a lock before you safely can make the call.

Much of what appears in this chapter is reference material. We suggest that you skim the material on the initial reading and refer back to specific sections as needed. However, we recommend that you read at least the mappings for exceptions, interfaces, and operations in detail because these sections cover how to call operations from a client, pass parameters, and handle exceptions.

In order to use the Java mapping, you should need no more than the Slice definition of your application and knowledge of the Java mapping rules. In particular, looking through the generated code in order to discern how to use the Java mapping is likely to be inefficient, due to the amount of detail. Of course, occasionally, you may want to refer to the generated code to confirm a detail of the mapping, but we recommend that you otherwise use the material presented here to see how to write your client-side code.

The `Ice` Package
All of the APIs for the Ice run time are nested in the `Ice` package, to avoid clashes with definitions for other libraries or applications. Some of the contents of the `Ice` package are generated from Slice definitions; other parts of the `Ice` package provide special-purpose definitions that do not have a corresponding Slice definition. We will incrementally cover the contents of the `Ice` package throughout the remainder of the manual.

Topics