


Streaming Interfaces



Ice provides convenient interfaces for streaming Slice types to and from a sequence of bytes. You can use these interfaces in many situations, such as when serializing types for persistent storage, and when using Ice's [dynamic invocation and dispatch facility](#).

The streaming interfaces are not defined in Slice, but are rather a collection of native classes provided by each language mapping.

 The streaming interfaces are currently supported in C++, Java, JavaScript and .NET.

There are two primary abstract classes: `InputStream` and `OutputStream`. As you might guess, `InputStream` is used to extract Slice types from a sequence of bytes, while `OutputStream` is used to convert Slice types into a sequence of bytes. The classes provide the functions necessary to manipulate all of the core Slice types:

- Primitives (`bool`, `int`, `string`, etc.)
- Sequences of primitives
- Proxies
- Objects

The classes also provide functions that handle various details of the [Ice encoding](#). Using these functions, you can manually insert and extract constructed types, such as dictionaries and structures, but doing so is tedious and error-prone. To make insertion and extraction of constructed types easier, the Slice compilers generate type-specific code that manages the low-level details for you.

The remainder of this section describes the streaming interfaces for each supported language mapping. To properly use the streaming interfaces, you should be familiar with the [Ice encoding](#). For an example that demonstrates the use of the streaming interfaces, refer to the `Ice/invoke` directory in the Ice demo distribution.

Topics

- [C++ Streaming Interfaces](#)
- [Java Streaming Interfaces](#)
- [Java Compat Streaming Interfaces](#)
- [C-Sharp Streaming Interfaces](#)
- [JavaScript Streaming Interfaces](#)

[Back to Top ^](#)

See Also

- [Dynamic Invocation and Dispatch](#)
- [Data Encoding](#)

