

Data Encoding

The key goals of the Ice data encoding are simplicity and efficiency. In keeping with these principles, the encoding does not align primitive types on word boundaries and therefore eliminates the wasted space and additional complexity that alignment requires. The Ice data encoding simply produces a stream of contiguous bytes; data contains no padding bytes and need not be aligned on word boundaries.

Data is always encoded using little-endian byte order for numeric types. (Most machines use a little-endian byte order, so the Ice data encoding is "right" more often than not.) Ice does not use a "receiver makes it right" scheme because of the additional complexity this would introduce. Consider, for example, a chain of receivers that merely forward data along the chain until that data arrives at an ultimate receiver. (Such topologies are common for event distribution services.) The Ice protocol permits all the intermediates to forward the data without requiring it to be unmarshaled: the intermediates can forward requests by simply copying blocks of binary data. With a "receiver makes it right" scheme, the intermediates would have to unmarshal and remarshal the data whenever the byte order of the next receiver in the chain differs from the byte order of the sender, which is inefficient.

Ice requires clients and servers that run on big-endian machines to incur the extra cost of byte swapping data into little-endian layout, but that cost is insignificant compared to the overall cost of sending or receiving a request.

Topics

- [Basic Data Encoding](#)
- [Data Encoding for Exceptions](#)
- [Data Encoding for Classes](#)
- [Data Encoding for Interfaces](#)
- [Data Encoding for Proxies](#)
- [Data Encoding for Optional Values](#)

See Also

- [Protocol Messages](#)