Stream Helper Functions in Java

The stream classes provide all of the low-level functions necessary for encoding and decoding Ice types. However, it would be tedious and error-prone to manually encode complex Ice types such as classes, structs, and dictionaries using these low-level functions. For this reason, the Slice compiler optionally generates helper functions for streaming complex Ice types.

We will use the following Slice definitions to demonstrate the language mapping:

Slice

```
module M {
    sequence<...> Seq;
    dictionary<...> Dict;
    struct S {
        ...
    };
    enum E { ... };
    class C {
        ...
    };
};
```

The Slice compiler generates the corresponding helper functions shown below:

Java

```
package M;

public class SeqHelper {
    public static T[] read(Ice.InputStream in);
    public static void write(Ice.OutputStream out, T[] v);
}

public class DictHelper {
    public static java.util.Map read(Ice.InputStream in);
    public static void write(Ice.OutputStream out, java.util.Map<..., ...> v);
}

public class SHelper {
    public static S read(Ice.InputStream in);
    public static void write(Ice.OutputStream out, S v);
}

public class EHelper {
    public static E read(Ice.InputStream in);
    public static void write(Ice.OutputStream out, E v);
}

public class CHelper {
    public static void read(Ice.InputStream in, CHolder h);
    public static void write(Ice.OutputStream out, C v);
}

public class CPrxHelper {
    public static CPrx read(Ice.InputStream in);
    public static void write(Ice.OutputStream out, CPrx v);
}
```

In addition, the Slice compiler generates the following member functions for `struct` and `enum` types:

Java

```
public class S ... {
    ...
    public void ice_read(Ice.InputStream in);
    public void ice_write(Ice.OutputStream out);
};
public class E... {
    ...
    public void ice_read(Ice.InputStream in);
    public void ice_write(Ice.OutputStream out);
}
```

Be aware that a call to `CHelper.read` does not result in the immediate extraction of an Ice object. The `value` member of the given `CHolder` object is updated when `readPendingObjects` is invoked on the input stream.

See Also

- Data Encoding
- Using the Slice Compiler for Java
- The InputStream Interface in Java