20.1 Input and Output

Java provides *I/O streams* as a general mechanism for dealing with data input and output. I/O streams implement *sequential processing* of data. An *input stream* allows an application to read a sequence of data, and an *output stream* allows an application to write a sequence of data. An *input stream* acts as a *source* of data, and an *output stream* acts as a *destination* of data. The following entities can act as both input and output streams:

- A file—which is the focus in this chapter
- An array of bytes or characters
- A network connection

There are two categories of I/O streams:

- *Byte streams* that process *bytes* as a unit of data
- *Character streams* that process *characters* as a unit of data

A *low-level I/O stream* operates directly on the data source (e.g., a file or an array of bytes), and processes the data primarily as *bytes*.

A *high-level I/O stream* is *chained* to an underlying stream, and provides additional capabilities for processing data managed by the underlying stream—for example, processing bytes from the underlying stream as Java primitive values or objects. In other words, a high-level I/O stream acts as a *wrapper* for the underlying stream.

In the rest of this chapter we primarily explore how to use I/O streams of the standard I/O API provided by the `java.io` package to read and write various kinds of data that is stored in files.