1.3. FLUJOS O STREAMS. TIPOS

El sistema de entrada/salida en Java presenta una gran cantidad de clases que se implementan en el paquete **java.io**. Usa la abstracción del flujo (**stream**) para tratar la comunicación de información entre una fuente y un destino; dicha información puede estar en un fichero en el disco duro, en la memoria, en algún lugar de la red, e incluso en otro programa. Cualquier programa que tenga que obtener información de cualquier fuente necesita abrir un stream, igualmente si necesita enviar información abrirá un stream y se escribirá la información en serie. La vinculación de este stream al dispositivo físico la hace el sistema de entrada y salida de Java. Se definen dos tipos de flujos:

- **Flujos de bytes (8 bits):** realizan operaciones de entradas y salidas de bytes y su uso está orientado a la lectura/escritura de datos binarios. Todas las clases de flujos de bytes descienden de las clases **InputStream** y **OutputStream**, cada una de estas clases tienen varias subclases que controlan las diferencias ente los distintos dispositivos de entrada/salida que se pueden utilizar.
- **Flujos de caracteres (16 bits):** realizan operaciones de entradas y salidas de caracteres. El flujo de caracteres viene gobernado por las clases **Reader** y **Writer**. La razón de ser de estas clases es la internacionalización; la antigua jerarquía de flujos de E/S solo soporta flujos de 8 bits no manejando caracteres Unicode de 16 bits que se utilizaba con fines de internacionalización.

1.3.1. Flujos de bytes (Byte streams)

La clase **InputStream** representa las clases que producen entradas de distintas fuentes, estas fuentes pueden ser: un array de bytes, un objeto String, un fichero, una "tubería" (se ponen los elementos en un extremo y salen por el otro), una secuencia de otros flujos, otras fuentes como una conexión a Internet, etc. Los tipos de **InputStream** se resumen en la siguiente tabla:

CLASE	Función
`ByteArrayInputStream`	Permite usar un espacio de almacenamiento intermedio de memoria
`StringBufferInputStream`	Convierte un String en un **InputStream**
`FileInputStream`	Flujo de entrada hacia fichero, lo usaremos para leer información de un fichero
`PipedInputStream`	Implementa el concepto de "tubería"
`FilterInputStream`	Proporciona funcionalidad útil a otras clases **InputStream**
`SequenceInputStream`	Convierte dos o más objetos **InputStream** en un **InputStream** único

Los tipos de **OutputStream** incluyen las clases que deciden dónde irá la salida: a un array de bytes, un fichero o una "tubería". Se resumen en la siguiente tabla:

CLASE	Función
`ByteArrayOutputStream`	Crea un espacio de almacenamiento intermedio en memoria. Todos los datos que se envían al flujo se ubican en este espacio
`FileOutputStream`	Flujo de salida hacia fichero, lo usaremos para enviar información a un fichero
`PipedOutputStream`	Cualquier información que se desee escribir aquí acaba automáticamente como entrada del **PipedInputStream** asociado. Implementa el concepto de "tubería"
`FilterOutputStream`	Proporciona funcionalidad útil a otras clases **OutputStream**

La Figura 1.1 muestra la jerarquía de clases para lectura y escritura de flujos de bytes.

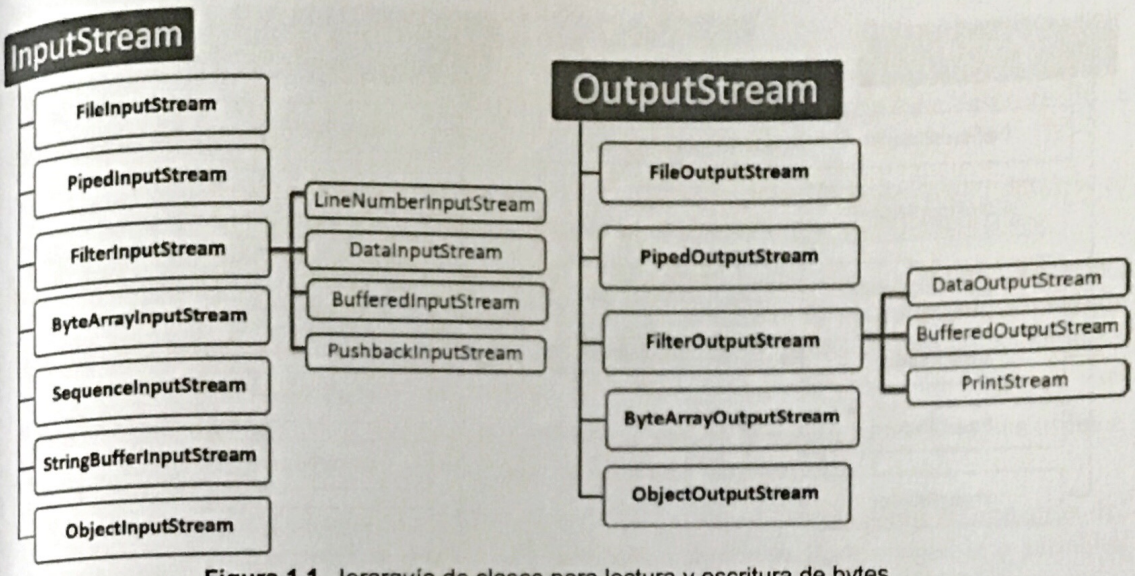

Figura 1.1. Jerarquía de clases para lectura y escritura de bytes.

Dentro de los flujos de bytes están las clases **FileInputStream** y **FileOutputStream** que manipulan los flujos de bytes provenientes o dirigidos hacia ficheros en disco y se estudiarán en los siguientes apartados.

1.3.2. Flujos de caracteres (Character streams)

Las clases **Reader** y **Writer** manejan flujos de caracteres Unicode. Hay ocasiones en las que hay que usar las clases que manejan bytes en combinación con las clases que manejan caracteres. Para lograr esto hay clases "puente" (es decir, convierte los streams de bytes a streams de caracteres): **InputStreamReader** que convierte un **InputStream** en un **Reader** y **OutputStreamWriter** que convierte un **OutputStream** en un **Writer** (convierte streams de caracteres a streams de bytes).

La siguiente tabla muestra la correspondencia entre las clases de flujos de bytes y de caracteres:

CLASES DE FLUJOS DE BYTES	CLASE CORRESPONDIENTE DE FLUJO DE CARACTERES
`InputStream`	`Reader`, convertidor `InputStreamReader`
`OutputStream`	`Writer`, convertidor `OutputStreamReader`
`FileInputStream`	`FileReader`
`FileOutputStream`	`FileWriter`
`StringBufferInputStream`	`StringReader`
(sin clase correspondiente)	`StringWriter`
`ByteArrayInputStream`	`CharArrayReader`
`ByteArrayOutputStream`	`CharArrayWriter`
`PipedInputStream`	`PipedReader`
`PipedOutputStream`	`PipedWriter`

La Figura 1.2 muestra la jerarquía de clases para lectura y escritura de flujos de caracteres.

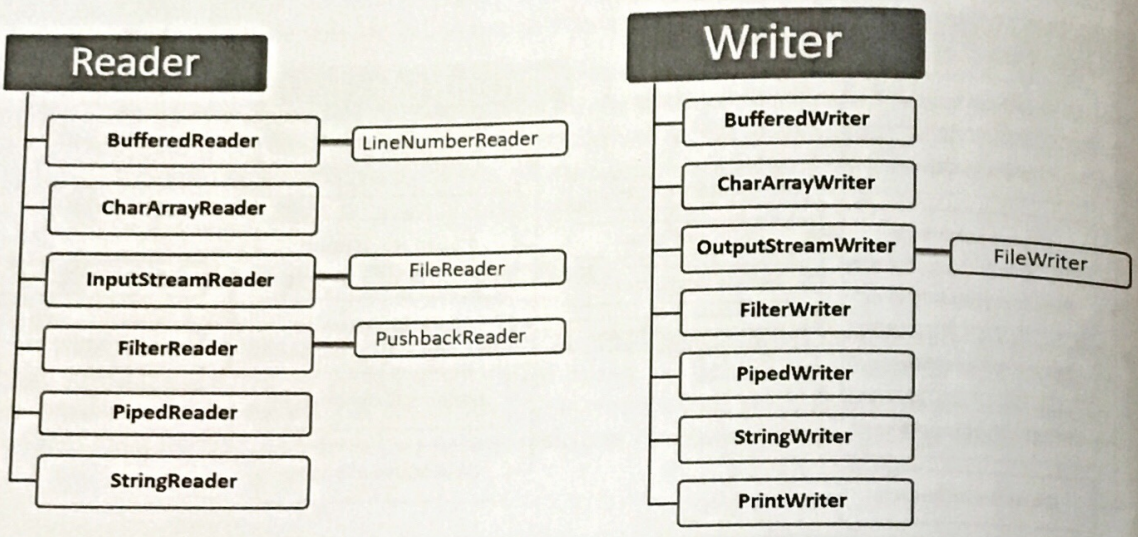

Figura 1.2. Jerarquía de clases para lectura y escritura de flujos de caracteres.

Las clases de flujos de caracteres más importantes son:

- Para acceso a ficheros, lectura y escritura de caracteres en ficheros: **FileReader** y **FileWriter**.
- Para acceso a caracteres, leen y escriben un flujo de caracteres en un array de caracteres: **CharArrayReader** y **CharArrayWriter**.
- Para buferización de datos: **BufferedReader** y **BufferedWriter**, se utilizan para evitar que cada lectura o escritura acceda directamente al fichero, ya que utilizan un buffer intermedio entre la memoria y el stream.

1.4. FORMAS DE ACCESO A UN FICHERO

Hay dos formas de acceso a la información almacenada en un fichero: acceso secuencial y acceso directo o aleatorio: