10.A. Introducción al almacenamiento de datos y flujos.

4. Clases relativas a flujos.

Existen dos tipos de flujos, flujos de bytes (byte streams) y flujos de caracteres (character streams).

- Los **flujos de caracteres** (16 bits) se usan para manipular datos legibles por humanos (por ejemplo un fichero de texto). Vienen determinados por dos clases abstractas: `Reader` y `Writer`. Dichas clases manejan flujos de caracteres Unicode. De ellas derivan subclases concretas que implementan los métodos definidos destacados los métodos `read()` y `write()` que, en este caso, leen y escriben caracteres de datos respectivamente.

- Los **flujos de bytes** (8 bits) se usan para manipular datos binarios, legibles solo por la maquina (por ejemplo un fichero de programa). Su uso está orientado a la lectura y escritura de datos binarios. El tratamiento del flujo de bytes viene determinado por dos clases abstractas que son `InputStream` y `OutputStream`. Estas dos clases son las que definen los métodos que sus subclases tendrán implementados y, de entre todos, destacan `read()` y `write()` que leen y escriben bytes de datos respectivamente.

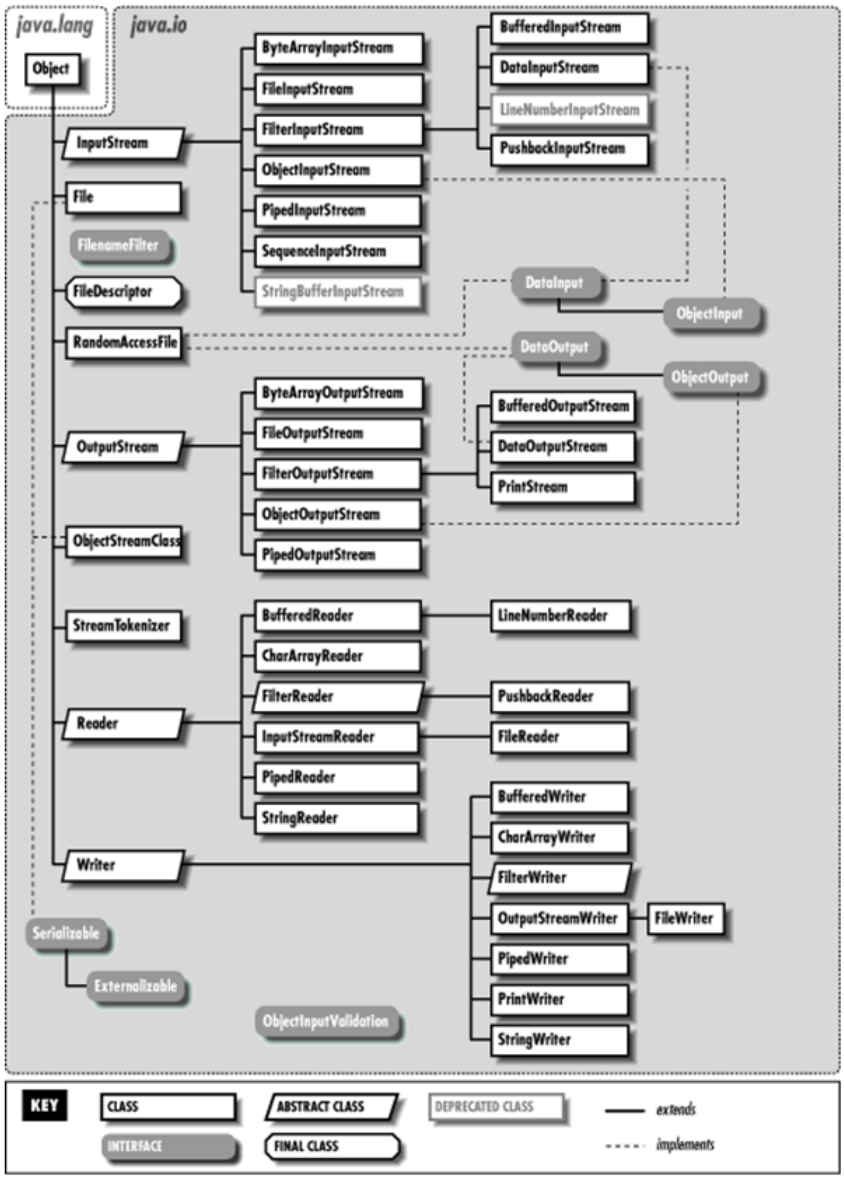

Imagen extraída de curso Programación del MECD.

Las clases del paquete `java.io` se pueden ver en la ilustración. Destacamos las clases relativas a flujos:

- `BufferedInputStream`: permite leer datos a través de un flujo con un buffer intermedio.

- `BufferedOutputStream`: implementa los métodos para escribir en un flujo a través de un buffer.
- `FileInputStream`: permite leer bytes de un fichero.
- `FileOutputStream`: permite escribir bytes en un fichero o descriptor.
- `StreamTokenizer`: esta clase recibe un flujo de entrada, lo analiza (parse) y divide en diversos pedazos (tokens), permitiendo leer uno en cada momento.
- `StringReader`: es un flujo de caracteres cuya fuente es una cadena de caracteres o string.
- `StringWriter`: es un flujo de caracteres cuya salida es un buffer de cadena de caracteres, que puede utilizarse para construir un string.

Destacar que hay clases que se **"montan" sobre otros flujos para modificar la forma de trabajar con ellos.** Por ejemplo, con `BufferedInputStream` podemos añadir un buffer a un flujo `FileInputStream`, de manera que se mejore la eficiencia de los accesos a los dispositivos en los que se almacena el fichero con el que conecta el flujo.