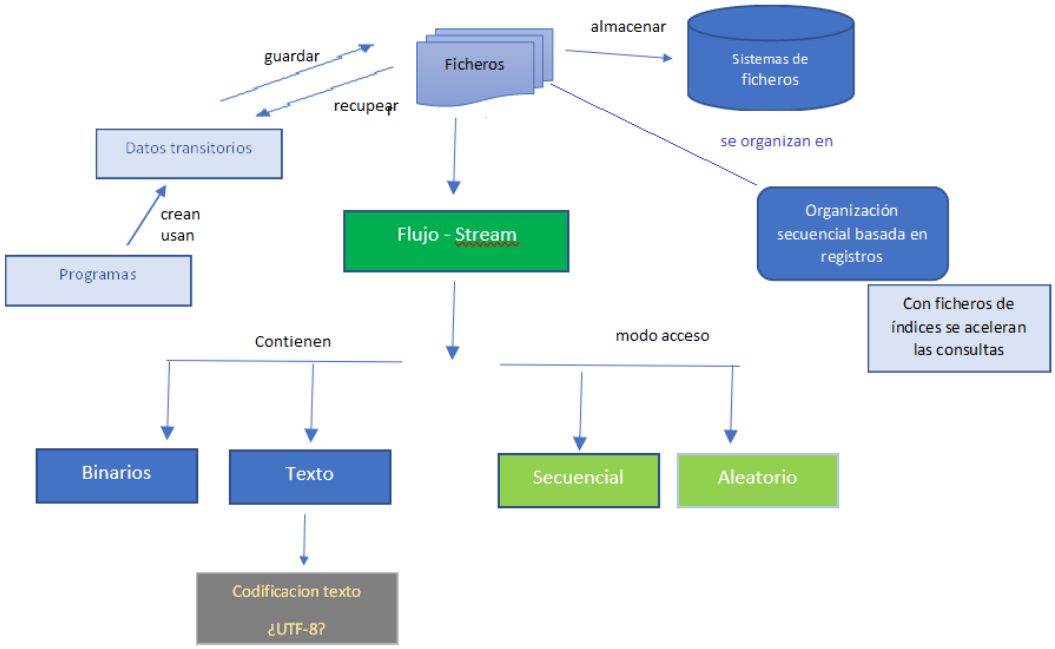
AD1. Ficheros.

El tema AD1 lo vamos a dedicar a los ficheros, repasaremos conceptos vistos en el módulo profesional de Programación por su relevancia para el acceso a datos.



Tipos de ficheros.

Los ficheros podemos clasificarlos en dos grandes grupos:

1. Ficheros de texto.
2. Ficheros binarios. Son los que contienen imágenes, videos, ficheros. También se pueden considerar en este tipo los propios de Word, porque incluyen más información que solo el texto (gráficos, imágenes, etc.).

Ejercicio: completar las siguientes tablas.

Ejemplos de ficheros de texto:

.txt	Fichero de texto plano
.xml	
.json	
.conf, .props	
.sql	
.html, .css	

Ejemplos de ficheros binarios:

.pdf	
.jpg	
.avi	
.doc, .ppt	

Propiedades de un fichero con System Properties

En JAVA, podemos averiguar las propiedades de un fichero utilizando las System Properties.

Ejercicio: Realiza un programa en JAVA para mostrar por pantalla algunas "keys" interesantes de System Properties, por ejemplo, el directorio del usuario, el nombre del usuario, el directorio de trabajo, etc.

La clase FILE en JAVA

La página oficial de Java contiene información sobre la clase FILE.

Recuerda que este curso vamos a emplear Java 17. Visita el enlace:

https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/io/File.html

EXCEPCIONES: Captura y tratamiento.

Excepción: Es un evento que ocurre durante la ejecución de un programa e interrumpe su curso normal. Por ejemplo, dividir por cero.

La gestión de las excepciones en Java es un factor fundamental para el éxito de un programa.

Las excepciones no gestionadas, normalmente abortan el programa o generan un mensaje de error.

La captura de las excepciones permite decidir cómo se gestiona esa interrupción para que el programa sea fiable.

La captura de excepciones suele hacerse fácilmente con el bloque try-catch. Si tenemos varias excepciones, lo más normal es tratarlas de una en una y agruparlas bajo exception solo en el caso de querer capturar alguna que se escapa de las anteriores.

Actividad 1: Construye un pequeño programa que divida dos números, cuando el denominador es 0. Trata la excepción "ArithmeticException" capturándola con try-catch.

Consejo: Los mensajes de error es mejor escribirlos con System.err que con System.out.

Recuerda: Las excepciones son objetos de Java de la clase Exception o de alguna subclase de ella.

Declarar excepciones lanzadas por un método de clase (Trows IOExcpetion)

Las excepciones de un método de una clase que el compilador no es capaz de tratar con try-catch, generará error. En ese caso, la excepción se eleva a la clase que llamó al método, empleando throws IOException.

Excepciones, inicialización y liberación de recursos (try-catch-finally)

En java el habitual el uso de recursos, por ejemplo, flujos a ficheros:

Inicializar y asignar recursos. Cuerpo: uso de recursos Finalizar y liberar recursos.

En estas estructuras, el tratamiento de excepciones será:

Inicializar y asignar recursos.

```
Try {
        Cuerpo: uso de recursos
} catch (Tipo1_Exception e1) {}
} catch (Tipo21_Exception e2) {}
Finally {
    Finalizar y liberar recursos}
```

Acceso secuencial a ficheros en Java.

Las operaciones sobre ficheros en java las realizamos con flujos (streams) mediante las clases del paquete java.io.

Hay una jerarquía de clases que deriva de las cuatro posibles combinaciones de flujos:

- Flujos binarios y flujos de texto.
- Flujos de entrada y flujos de salida.

Un Stream es la serialización de un fichero:

- secuencia ordenada de datos con un determinado origen y destino
- su origen o destino puede ser un fichero, pero también un string o un dispositivo (p.e. el teclado)

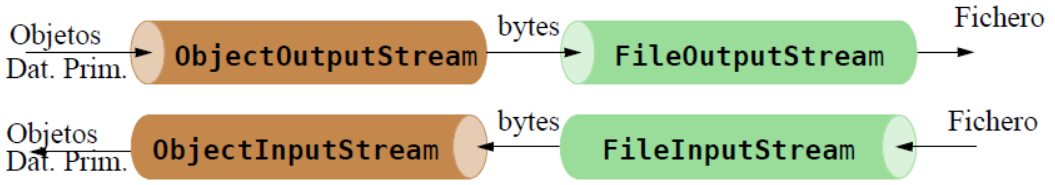
Para poder usar un stream primero hay que abrirle

- se abre en el momento de su creación
- y hay que cerrarle cuando se deja de utilizar

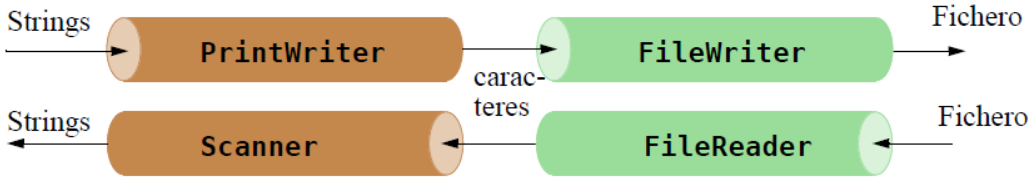
Las clases relacionadas con streams se encuentran definidas en el paquete java.io (io es la abreviatura de Input/Output)

Los **Streams** normalmente se utilizan por parejas formadas por un stream iniciador y un filtro:

Binarios



De Texto:

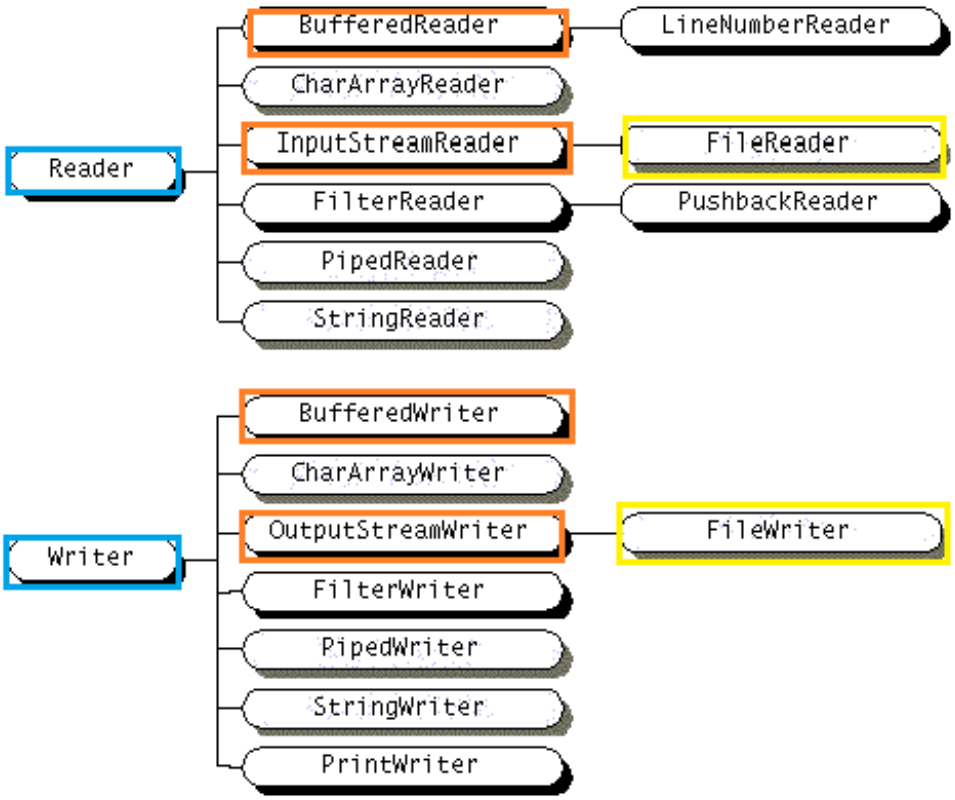


Jerarquías de clase:

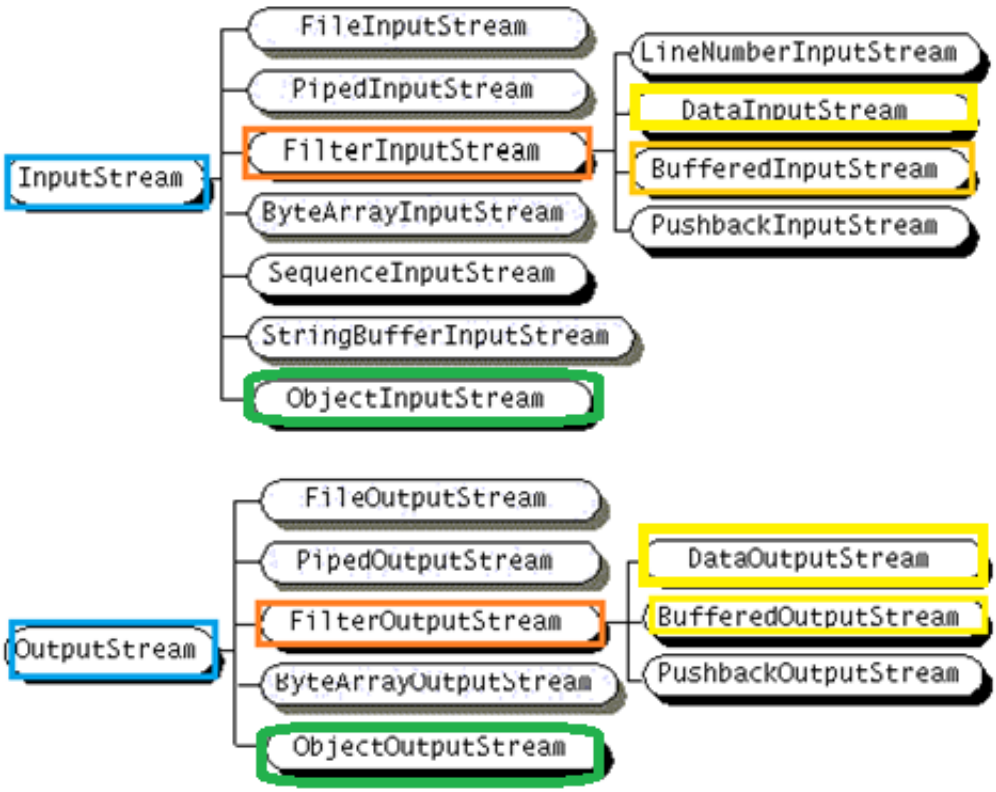
Para leer y escribir flujos, Java proporciona dos jerarquías de clases:

- Para flujos de texto
- Para flujos binarios.

Esquema de flujos de texto

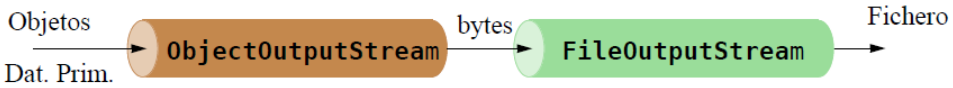


Esquema de flujos binarios:



Escribir en ficheros binarios:

Se usa la pareja de inciador y filtro:



Esquema general de uso:

```
ObjectOutputStream out = null;
try {
  // crea los streams y los conecta
  out = new ObjectOutputStream(new FileOutputStream(nomFich));
  // escribe en el fichero
  ... diferente en cada caso ...
} finally {
  if (out != null)
    out.close();
}
```

Operaciones con ficheros de acceso aleatorio.

En Java para el acceso aleatorio se utiliza la clase RandomAccessFile que aporta:

- La función seek(). Se encarga de situar el puntero en cualquier posición.
- Se pueden realizar operaciones tanto de lectura como de escritura sobre el fichero.

Limitaciones:

No es posible eliminar o insertar bloques de bytes o de caracteres en mitad del fichero. Tampoco es posible reemplazar un bloque de un fichero por otro, salvo que tengan exactamente el mismo tamaño en bytes.

La forma de sortear estas limitaciones es empleando ficheros temporales auxiliares.