■ 1. Ficheros básicos (java.io)

Clase File → Info y gestión de ficheros/directorios

```
File f = new File("ruta/fichero.txt");
```

• No crea el fichero automáticamente.
• Métodos útiles: exists(), createNewFile(), delete(), mkdir(), length(), getName(), getPath(), isFile(), isDirectory().
■ Permite manipular rutas y ficheros pero no leer ni escribir contenido directamente.

■ 2. Streams (Flujos)

Bytes (InputStream/OutputStream) → datos binarios

Caracteres (Reader/Writer) →texto

FileReader/FileWriter → texto

BufferedReader → leer líneas completas:

```
BufferedReader br = new BufferedReader(new FileReader("f.txt"));
String linea = br.readLine();
```

■ Más eficiente que leer carácter a carácter.

✍■ 3. Ficheros de acceso aleatorio

```
RandomAccessFile raf = new RandomAccessFile("datos.dat", "rw");
```

```
StringBuffer s= StringBuffer(String)
```

```
s.setLenght(longitud);
```

```
para acotar los strings en
```

• "r" solo lectura, "rw" lectura/escritura
• seek(pos), getFilePointer(), length(), skipBytes(n), read(), write()

- Acceso directo sin recorrer todo el fichero.

Tipos y tamaños: int=4, double=8, char=2, short=2, byte=1, float=4.

4. XML – DOM

DOM = Document Object Model (todo XML cargado en memoria). `DocumentBuilderFactory f = DocumentBuilderFactory.newInstance(); DocumentBuilder b = f.newDocumentBuilder(); Document doc = b.parse(new File("libros.xml"));`

- getDocumentElement(), getElementsByTagName(), getTextContent()
- appendChild(), removeChild()

Guardar cambios: `Transformer t = TransformerFactory.newInstance().newTransformer();`

- Ideal para modificar XML pequeños.

5. XML – SAX

SAX = lectura secuencial, no carga en memoria. `SAXParserFactory f = SAXParserFactory.newInstance(); SAXParser p = f.newSAXParser(); p.parse(new File("libros.xml"), new ManejadorSAX());`

- Ideal para leer XML grandes sin modificarlos.

6. XML – JAXB

JAXB = mapeo XML ↔ Objetos Java (alto nivel).

tags: @XmlRootElement(nombre) @XmlType(propOrder{“orden”}) @XmlElementWrapper(nombre)

@XmlElement(nombre) @XmlAttribute(nombre)

```
JAXBContext c = JAXBContext.newInstance(ObjetoPadre.class);
```

```
Unmarshaller u = c.createUnmarshaller();
ObjetoPadre miObjeto = (Objeto) u.unmarshal(new File("Archivo.xml"));
Al crear los sysos se trabaja sobre la clase padre nueva.


Marshaler es lo mismo pero con c.createMarshaller()
Se declaran todas las cosas del objeto (array y objetos del array incluidos)
y se añade: m.setProperty(m.JAXB_FORMATTED_OUTPUT, Boolean.TRUE) para dar formato xml
despues simplemente m.marshal(ObjetoPadreNuevo, archivo.xml)
```

- Unmarshal = XML → objeto
- Marshal = objeto → XML
  - Ideal para des/serializar sin procesar estructura manualmente.

■ Extra útil

try-with-resources:

```
try (BufferedReader br = new BufferedReader(new FileReader("f.txt"))) { // auto-cierra }
```

- Rutas relativas → desde proyecto
- Rutas absolutas → empiezan con / o C:/