

## IES CHAN DO MONTE

### C.S. de Desarrollo de Aplicaciones Multiplataforma

#### Modulo Acceso a datos

### ACTIVIDAD 6: Persistencia XML (JAXB)

Esta actividad tiene como objetivo aplicar la API JAXB (Java Architecture for XML Binding) para la lectura y escritura de documentos XML en el contexto de vinculación por clases. Se trabajará con los documentos corredores.xml y equipos.xml, reutilizando las clases del paquete Clases y las utilidades de marshalling/unmarshalling definidas en XMLJAXBUtils.java.

#### Objetivos

- Consolidar el uso de JAXB para vincular documentos XML con clases Java anotadas.
- Aplicar el proceso de unmarshalling para convertir XML en objetos Java.
- Aplicar el proceso de marshalling para generar documentos XML a partir de objetos Java.
- Mantener la arquitectura por capas, separando la lógica de negocio de la persistencia.

Esta práctica se plantea como una continuación del desarrollo de la aplicación de gestión de corredores, reutilizando las clases ya creadas en el paquete Clases (Corredor, Velocista, Fondista, Puntuación, Equipo, Patrocinador) y manteniendo la arquitectura por capas establecida.

Con el objetivo de aplicar el modelo **JAXB** para la vinculación de documentos XML a objetos y viceversa, se creará un nuevo paquete llamado **PersistenciaJAXB**.

- Dentro de este paquete se implementarán **las clases de acceso a los documentos XML mediante JAXB**:

**CorredoresJAXB.java**

**EquiposJAXB.java**

**XMLJAXBUtils.java** (clase de utilidades con métodos genéricos de marshalling y unmarshalling).

- Crear dentro de PersistenciaJAXB un subpaquete llamado **Clases**.

Dentro de este subpaquete, organizar dos carpetas:

- **Corredores** → incluir las clases necesarias para la vinculación de corredores.
- **Equipos** → incluir las clases necesarias para la vinculación de equipos.
- Se pueden **reutilizar las clases anteriores** ya definidas en el proyecto, adaptándolas con anotaciones JAXB (@XmlRootElement, @XmlType, @XmlElement, @XmlAttribute, etc.) para que funcionen correctamente en el proceso de vinculación y añadiendo las necesarias para hacer la vinculación.

#### ✓ Definición de la Capa de Utilidad (XMLJAXBUtils)

- Implementar métodos genéricos para cargar documentos XML en objetos Java (*unmarshal*) y para guardar objetos Java en XML (*marshal*).
- Los métodos deben ser reutilizables para cualquier clase raíz vinculada.

Método	Propósito
<code>public static &lt;T&gt; void marshal(T objeto, String rutaArchivo) throws JAXBException</code>	Convierte el objeto Java (que debe ser la clase raíz, en un documento XML.
<code>public static &lt;T&gt; T unmarshal(Class&lt;T&gt; clase, String rutaArchivo) throws JAXBException</code>	Lee el documento XML de la ruta y lo convierte en una instancia del objeto Java de la clase raíz (Class<T>) indicada.

#### ✓ Definición del adaptador de fechas en formato localdate.

- Crear un adaptador que convierta entre LocalDate y su representación en XML (yyyy-MM-dd).
- Aplicar el adaptador en las clases que contengan fechas de nacimiento o fechas de donación.

## **Actividades prácticas**

### **1. Listar todos los equipos con JAXB**

- Mostrar la información de cada equipo y la información de sus patrocinadores (nombre, donación y fecha).

### **2. Listar todos los corredores con JAXB**

- Recorrer la lista de objetos Corredor obtenida y mostrar por consola:
- El tipo de corredor (Velocista o Fondista).
- El nombre y la fecha de nacimiento.
- Los atributos específicos de cada tipo (velocidad media o distancia máxima).
- La información de su historial de puntuaciones: **año y valor**.

### **3. Escribir corredores en un archivo nuevo**

- Crear en memoria varios corredores (fondista y velocistas) y encapsularlos en un objeto raíz Corredores
- Almacenarlos en un fichero nuevo CorredoresNuevos.xml

### **4. Añadir equipos nuevos provenientes de un fichero de texto.**

- Lee el fichero de texto original con equipos JAXB.
- Cada línea del fichero representa un equipo y contiene:  
`<código>, <nombre>, Patrocinadores: <patrocinador1>;<patrocinador2> ;...;`  
`<patrocinadorN>`
- Añade equipos nuevos
- Comprueba duplicados:
  1. Si código repetido → ✗ no se inserta el equipo.
  2. Si nombre repetido → ✗ no se inserta el equipo.
  3. Si patrocinador repetido → ⚠ sí se inserta el equipo, pero solo con el primer patrocinador que aparece en el XML.
- Visualiza mensajes indicando claros el error
- Guarda los cambios en EquiposUpdateJAXB.xml usando JABX.

Ejemplo de fichero de texto:

E9, Celta, Patrocinadores: Nike|1500.0|2025-01-15;Adidas|2500.0|2024-03-15

E10, Deportivo, Patrocinadores: Abanca|3000.0|2025-02-10

E11, Lugo, Patrocinadores: Gadis|1200.0|2025-03-05;Estrella Galicia|2000.0|2025-04-01;Nike|1800.0|2025-05-10

E5, Castilblanco, Patrocinadores: SponsorX|1000.0|2025-06-01

E12, Celta, Patrocinadores: SponsorY|1200.0|2025-07-01

E13, Ourense, Patrocinadores: Nike|1500.0|2025-01-15;Nike|2000.0|2025-02-20;Decathlon|800.0|2025-03-10