JAXB

Java Architecture for XML Binding

JAXB

Java JAXB o Java XML API Binding nos permite trabajar **con XML** de una forma cómoda usando Java. Vamos a ver una introducción a este estándar y sus anotaciones.

JAXB (Java Architecture for XML Binding)

JAXB (Java Architecture for XML Binding) es un estándar Java para transformar un esquema XML (o XSD) en una representación a objetos java. Mediante la API de JAXB podemos mapear un objeto Java a un documento XML ("marshall") y el proceso contrario, es decir, a partir de un esquema XML crear su conjunto de objeto Java asociado ("unmarshall").

CONCEPTOS

Parsear un documento XML consiste en "escanear" el documento y dividirlo o separarlo lógicamente en piezas discretas. El contenido parseado está entonces disponible para la aplicación.

Binding: Binding o vincular un esquema (schema) significa generar un conjunto de clases Java que representan el esquema.

Compilador de esquema o schema compiler: liga un esquema fuente a un conjunto de elementos de programa derivados. La vinculación se describe mediante un lenguaje de vinculación basado en XML.

JAXB

Binding runtime framework: proporciona operaciones de unmarshalling y marshalling para acceder, manipular y validar contenido XML usando un esquema derivado o elementos de programa.

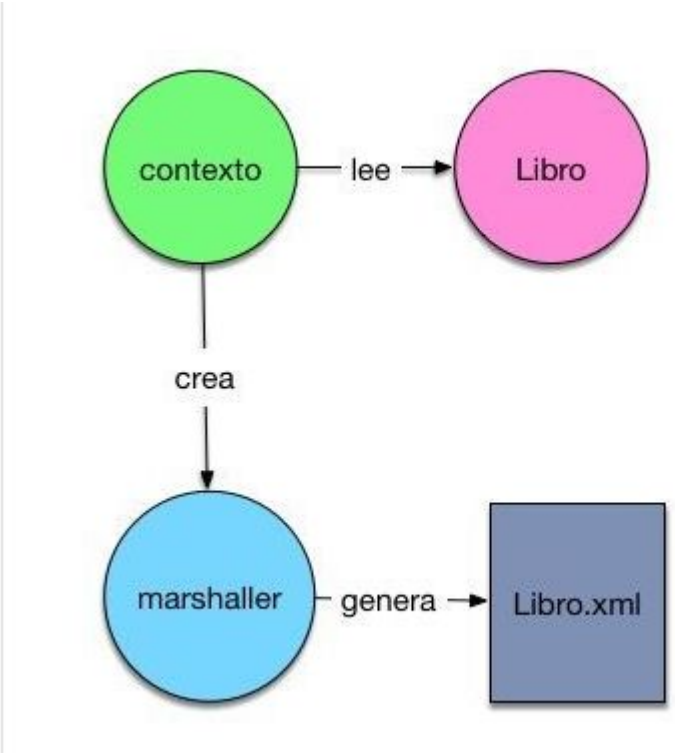
Marshalling: es un proceso de codificación de un objeto en un medio de almacenamiento, normalmente un fichero. Proporciona a una aplicación cliente la capacidad para **convertir un árbol de objetos Java JAXB a ficheros XML. P**or defecto, el marshaller usa codificación UTF-8 cuando genera los datos XML.

Unmarshalling: proporciona a una aplicación cliente la capacidad de convertir datos XML a objetos Java JAXB derivados.

JAXB - EJEMPLO

```
package com.arquitecturajava;

import javax.xml.bind.annotation.XmlElement;

import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement

public class Libro {

    private String titulo;

    private int paginas;

    public String getTitulo() {

        return titulo;

    }
```

JABX - EJEMPLO



JAXB - EJEMPLO

```
public Libro(String titulo, int paginas) {
        super();
        this.titulo = titulo;
        this.paginas = paginas;
    }
    public Libro() {
        super();
    }
```

JAXB - EJEMPLO

```
public void setTitulo(String titulo) {
        this.titulo = titulo;
    }
    @XmlElement(name="numeroPaginas")
    public int getPaginas() {
        return paginas;
    }
    public void setPaginas(int paginas) {
        this.paginas = paginas;
    }
```

JAXB - EJEMPLO

Hemos añadido dos anotaciones **@XmlRootElement** que especifica la clase raiz que vamos a convertir a XML . Por otro lado **@XmlElement** permite cambiar el nombre de los elementos cuando el fichero XML se construya. Es momento de generar el fichero XML .

JAXB - EJEMPLO

```
import javax.xml.bind.JAXBContext;

import javax.xml.bind.JAXBException;

import javax.xml.bind.Marshaller;

import javax.xml.bind.PropertyException;
```

JAXB - EJEMPLO - marshaller

```
public static void main(String[] args) {
    try {
        Libro libro= new Libro("Odisea 2001",400);
JAXBContext contexto = JAXBContext.newInstance(  libro.getClass() );
    Marshaller marshaller = contexto.createMarshaller();
```

JAXB - EJEMPLO marshaller

```
marshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT,
   Boolean.TRUE);
                marshaller.marshal(libro, System.out);
        }
}
```

JAXB - EJEMPLO

```
Catch (PropertyException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (JAXBException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }


    }
```

JAXB - RESULTADO

```
] --- exec:3.1.0:exec (default-cli) @ jaxbmaven ---
Hello World!
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<libro>
    <numeroPaginas>400</numeroPaginas>
    <titulo>Odisea 2001</titulo>
</libro>
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  6.877 s
Finished at: 2023-10-09T07:09:57+02:00
------------------------------------------------------------------------
```

JAXB - EJEMPLO - Unmarshaller

```
public static void main(String[] args) {
        try {
            JAXBContext context = JAXBContext.newInstance( Libro.class );
            Unmarshaller unmarshaller = context.createUnmarshaller();
            Libro libro = (Libro)unmarshaller.unmarshal(
                new File("src/Libro.xml") );
                    System.out.println(libro.getTitulo());
                    System.out.println(libro.getPaginas());
```

JAXB - EJEMPLO - Unmarshaller

```
        } catch (JAXBException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }
```

jabx

@XmlRootElement(name = "empleado"). Define que el elemento raíz del objeto Java se llamará empleado. (<empleado>)

@XmlType(propOrder = { "dni", "nombre", "edad", "puesto" }). Define el orden de los elementos dentro de otro elemento.

@XmlElement(name = "cargo"). Sirve para cambiar el nombre de un elemento en el documento XML. Por defecto el elemento aparece con el nombre del atributo de la clase Java pero podemos cambiarlo con esta etiqueta. En el ejemplo se cambia el nombre puesto por cargo. Se puede utilizar en los atributos tipo lista para indicar cómo se llamará cada elemento de la lista.

@XmlElementWrapper(name = "empleados"). Para crear un wrapper llamado empleados que englobe la estructura XML de los objetos Empleado.

```
▼<empresa>
   <cif>A58818501</cif>
   <nombre>TECNOMUR S.L.</nombre>
  ▼<empleados>
    ▼<empleado>
       <dni>12345678C</dni>
       <nombre>Carlos Pérez Ruíz</nombre>
       <edad>29</edad>
       <cargo>administrador</cargo>
     </empleado>
    ▼<empleado>
       <dni>87654321C</dni>
       <nombre>Claudia Ortiz Zaldo</nombre>
       <edad>31</edad>
       <cargo>informática</cargo>
     </empleado>
   </empleados>
 </empresa>
```