UA02. PRIMER ACCESO A DATOS

1.3. Conversiones

En Java existen una infinidad de librerías que podemos añadir a nuestro proyecto para poder transformar nuestros XML a cualquier otro formato. En este apartado, explicaremos cómo transformar nuestros ficheros XML a un formato JSON.

El formato JSON

JSON: siglas para **Javascript Object Notation**. Es un tipo de archivo de formato de texto derivado de Javascript, bastante ligero y que almacena información estructurada. Es fácil de interpretar y generar, y se utiliza para transferir información entre un cliente y un servidor.

Más información <https://www.w3schools.com/js/js_json_intro.asp>

Los archivos JSON siguen una estructura basada en definición de objetos, asignando un atributo y un valor. Un fichero JSON es capaz de definir seis tipos de valores: string, número, objeto, arrays, true, false o null.

Veamos un ejemplo:

```
{
    "coche":{
        "marca": "Seat",
        "modelo": "Ibiza",
        "color": "rojo",
        "matriculacion": 2019
    },
    "coche":{
        "marca": "Ford",
        "modelo": "Focus",
        "color": "rojo",
        "matriculacion": 2019
    }
}
```

Para definir el JSON, se abren y cierran corchetes. Los objetos se declaran entre comillas y los diferentes objetos se separan con una coma. El nombre y el valor de cada pareja van separados entre dos puntos. Cada objeto se considera un **string**; en cambio, los valores de los atributos pueden ser de cada tipo permitido nombrado en el parágrafo anterior.

```
1 public static String XML_PRUEBA ="<coche><id>1</id><modelo>Ibiza
2 try {
3     //Creamos el objeto que nos ayudara a convertir el XML en JS
4     JSONObject json = XML.toJSONObject(XML_PRUEBA);
```

```
5       //Identamos el json, le damos formato
6       String jsonFormatado = json.toString();
7       System.out.println(jsonFormatado);
8   } catch (JSONException je) {
9       System.out.println(je.toString());
10  }
```

Para esta conversión tenemos que añadir la **librería Jackson** a nuestro proyecto.

Este sería un ejemplo sencillo de la implementación de esta librería:

1. En primer lugar, tenemos que definir un XML, en el ejemplo tenemos un modelo sencillo de XML.
2. A continuación, definiremos un nuevo objeto usando la clase **JSONObject** que se rellenará con la llamada al método **XML.toJSONObject()**. Este método nos permitirá transformar cualquier XML a JSON, pasándole el XML como string.
3. Una vez transformado a objeto JSON, podemos pasarlo a string usando el método **toString()** y ya tendríamos la transformación realizada.
4. Tenemos que tener en cuenta que nuestro XML debe estar bien estructurado y tiene que estar validado y sin ningún error. Si hay algún error, no se hará la transformación correctamente y saltará un error que lo controlaremos en el **catch**.

Conversión a Web

XSL (Extensible Stylesheet Language) es toda una familia de recomendaciones del World Wide Web Consortium para expresar hojas de estilo en lenguaje XML. Una hoja de estilo **XSL** describe el proceso de presentación a través de un pequeño conjunto de elementos **XML**. Esta hoja, puede contener elementos de reglas que representan a as reglas de construcción y elementos de reglas de estilo que representan a las reglas de mezcla de estilos. Comprobaremos como a partir de un fichero XML que contiene datos y otro XSL que contiene la presentación de esos datos se puede generar un fichero **HTML** usando el lenguaje **Java**.

```xml
<?xml version="1.0"?>
<listadealumnos>
<alumno>
<nombre>Juan</nombre>
<edad>19</edad>
</alumno>
<alumno>
<nombre>María</nombre>
<edad>20</edad>
</alumno>
</listadealumnos>
```

```xml

<?xml version="1.0" encoding='UTF-8'?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match='/'>
    <html><xsl:apply-templates /></html>
</xsl:template>
<xsl:template match='listadodealumnos'>
    <head><title>LISTADO DE ALUMNOS</title></head>
    <body>
    <h1>LISTA DE ALUMNOS</h1>
    <table border='1'>
    <tr><th>Nombre</th><th>Edad</th></tr>
    <xsl:apply-templates select='alumno' />
    </table>
    </body>
</xsl:template>
<xsl:template match='alumno'>
    <tr><xsl:apply-templates /></tr>
</xsl:template>
<xsl:template match='nombre|edad'>

```

```
    <td><xsl:apply-templates /></td>
</xsl:template>
</xsl:stylesheet>
```

Para realizar la transformación, se necesita un objeto **Transformer** que se obtiene creando una instancia de **TransformerFactory** y aplicando el método **newTransformer** a la fuente XSL.

```
Transformer transformer = TransformerFactory.newInstance().newTr
```

La transformación se consigue llamando al método **transform()**, pasándole los datos (XML) y el stream de salida (el fichero HTML):

```
transformer.transform(datos, result);
```