```java
package org.adrianl.xml;

import lombok.Getter;
import lombok.NonNull;
import org.jdom2.Document;
import org.jdom2.Element;
import org.jdom2.JDOMException;
import org.jdom2.filter.Filters;
import org.jdom2.input.SAXBuilder;
import org.jdom2.output.Format;
import org.jdom2.output.XMLOutputter;
import org.jdom2.xpath.XPathExpression;
import org.jdom2.xpath.XPathFactory;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

@Getter
public class JDOMController {
    //1º VARIABLES
    private static JDOMController controller;   //Se crea el controlador
    private final String uri;   //Para la dirreccion del archivo
    private Document data;  //Documento

    //2º SINGLETON Solo se creará una instancia de la clase    Obtiene una instancia del controlador
    private JDOMController(String uri) {
        this.uri = uri;
    }  //Para dar valor a la uri
    public static JDOMController getInstance(@NonNull String uri) {
        if (controller == null)
            controller = new JDOMController(uri);
        return controller;
    }

    //3º Carga los datos desde un fichero dada su URI
    public void loadData() throws IOException, JDOMException {
        // JDOM Document trabaja con DOM, SAX y STAX Parser Builder classes
        SAXBuilder builder = new SAXBuilder();  //VER SAXBUILDER
        File xmlFile = new File(this.uri);  //uri
        //this.data = (Document) builder.build(xmlFile);//----------uri
        this.data = (Document) builder.build(uri);  //Mete en el Document el sax builder con el archivo que tiene la uri
    }



    // 4º SE CRE EL PREPROCESADO
```

```
50      private XMLOutputter preProcess() {
51          XMLOutputter xmlOutput = new XMLOutputter();
52          xmlOutput.setFormat(Format.getPrettyFormat());
53          return xmlOutput;
54      }
55
56      //5º MUESTRA EL XML
57      public void printXML() throws IOException {
58          XMLOutputter xmlOutput = this.preProcess();
59          xmlOutput.output(this.data, System.out);
60      }
61
62      // 6º ESCRIBE EL XML EN OTRO ARCHIVO
63      public void writeXMLFile(String uri) throws IOException {
64          XMLOutputter xmlOutput = this.preProcess();
65          xmlOutput.output(this.data, new FileWriter(uri));
66          System.out.println("Fichero XML generado con éxito");
67      }
68
69      // 7º Se obtiene la lista de usuarios del DOM cargado
70      public List<Noticia> getNoticias() {
71          // Tomamos el elemento raiz y obtenemos sus hijos
72          Element root = (Element) this.data.getRootElement();
73          Element channel = root.getChild("channel");
74          List<Element> listOfNoticias = channel.getChildren("item");
75          // Lista de noticias
76          List<Noticia> noticiaList = new ArrayList<>();
77          // Por cada elemento
78          listOfNoticias.forEach(noticiaElement -> {
79              Noticia noticia = new Noticia();
80              //noticia.setId(Integer.parseInt(noticiaElement.getAttributeValue("id")));
81              noticia.setTitulo(noticiaElement.getChildText("title"));
82              noticia.setCategoria(noticiaElement.getChildText("category"));
83              noticia.setDescripcion(noticiaElement.getChildText("description"));
84              noticia.setFecha(noticiaElement.getChildText("pubDate"));
85              noticia.setEnlace(noticiaElement.getChildText("link"));
86              noticia.setAuthor(noticiaElement.getChildText("dc:creator"));
87              noticiaList.add(noticia);
88          });
89          return noticiaList;
90      }
91
92
93
94
95
96
97
98      //----------------------------------------8º Consultas con XPATH
    ------------------------------------------------
99
```

```
100     //Obtener todos los elementos nombre
101     public List<String> getAllTitulos() {
102         XPathFactory xpath = XPathFactory.instance();
103         XPathExpression<Element> expr = xpath.compile("//item/title", Filters.element
        ());
104         //XPathExpression<Attribute> expr = xpath.compile("//item/@id", Filters.
        attribute());
105         List<Element> noticias = expr.evaluate(this.data);
106         //List<Attribute> noticias = expr.evaluate(this.data);
107         List<String> titulos = new ArrayList<String>();
108         noticias.forEach(noticia -> titulos.add(noticia.getValue().trim()));
109         return titulos;
110     }
111 
112     //Obtiene el nombre del elemento con indice indicado El primero en este caso
113     public String getIndexTitulo(int index) {
114         XPathFactory xpath = XPathFactory.instance();
115         XPathExpression<Element> expr = xpath.compile("//item["+index+"]/title",
        Filters.element());
116         //XPathExpression<Attribute> expr = xpath.compile("//item[@id='"+id+"']/title",
        Filters.attribute());
117         Element title = expr.evaluateFirst(this.data);  //Element titulo = expr.
        evaluateFirst(uri);
118         return title.getValue();
119     }
120 
121     //---------------------------------------9º Operaciones CRUD
        ----------------------------------------------------
122 
123     //Creación del xml con datos cargados del Rss
124     public void initData() {
125         // Documento vacío
126         this.data = new Document();
127         // Nodo raíz
128         this.data.setRootElement(new Element("noticias"));//noticias
129     }
130 
131     //Añade un elemento de la clase Noticia
132     private static Element createNoticiaElement(Noticia noticia) {
133         Element noticiaElement = new Element("noticia");//Noticia
134         noticiaElement.addContent(new Element("title").setText(noticia.getTitulo()));
135         noticiaElement.addContent(new Element("category").setText(noticia.
        getCategoria()));
136         noticiaElement.addContent(new Element("description").setText(noticia.
        getDescripcion()));
137         noticiaElement.addContent(new Element("pubDate").setText(noticia.getFecha
        ()));
138         //noticiaElement.addContent(new Element("age").setText(Integer.toString(
        noticia.getAge())));
139         noticiaElement.addContent(new Element("link").setText(noticia.getEnlace()));
140         noticiaElement.addContent(new Element("creator").setText(noticia.getAuthor
```

```
140  ()));
141          return noticiaElement;
142      }
143
144      // Añade una noticia al DOM
145      public void addNoticia(Noticia noticia) {
146          Element root = (Element) this.data.getRootElement();
147          root.addContent(createNoticiaElement(noticia));
148      }
149
150      // Añade un nuevo elemento con un determinado tag al usuario
151      public void addElement(String tag, String value) {
152          Element rootElement = this.data.getRootElement();
153          List<Element> listNoticiasElement = rootElement.getChildren("noticia");
154          listNoticiasElement.forEach(noticiaElement -> {
155              noticiaElement.addContent(new Element(tag).setText(value));
156          });
157      }
158
159      //Elimina un elemento dado un tag Elimina el autor
160      public void deleteElement(String tag) {
161          Element rootElement = this.data.getRootElement();
162          List<Element> listNoticiasElement = rootElement.getChildren("noticia");
163          listNoticiasElement.forEach(noticiaElement -> {
164              String name = noticiaElement.getChildText("creator");
165              // if (name != null)
166              noticiaElement.removeChild(tag);
167          });
168      }
169
170      //Pasa a mayúsculas el texto de un elemento de texto tag
171      public void ValorEnMayuscula(String tag) {
172          Element rootElement = this.data.getRootElement();
173          List<Element> listNoticiasElement = rootElement.getChildren("noticia");
174          listNoticiasElement.forEach(noticiaElement -> {
175              String name = noticiaElement.getChildText("title");
176              if (name != null)
177                  noticiaElement.getChild("title").setText(name.toUpperCase());
178          });
179      }
180  }
```