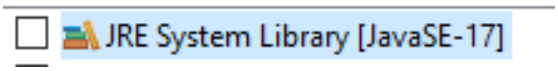
XML A SVG

El lenguaje que utilicé es Java en su versión 17.



El proyecto está organizado en dos paquetes: un paquete "file" en el que están las clases necesarias para leer y escribir ficheros y el paquete "main" en el que está la parte principal del código para convertir un XML a SVG. Para ejecutar el programa, en la clase Main.java se crea una instancia del método XMLtoSVG hay que pasar como parámetros el nombre del archivo de entrada y el de salida.

XMLtoSVG.java

Esta clase contiene otra clase, la clase Persona en la cual están los atributos de las personas y dos int y un boolean que se usan para crear el árbol.

Su método principal *crearSVG()* es el que es llamado por la clase Main y este llama en orden a los diferentes métodos que van creando el código del archivo SVG, llamando primero al método *inicio()* que escribe las primeras líneas del SVG.

Después llama al método *parse()* que es fundamental. Primero crea una conexión al archivo XML usando la clase FileInputStream, después se obtiene una nueva instancia de un DocumentBuilderFactory para seguidamente conseguir una instancia de DocumentBuilder que usamos para poder conseguir un DOM (Document) de nuestro archivo XML. Tras esto, debemos crear una instancia de XPath que usaremos un poco más adelante.

```
FileInputStream XML = new FileInputStream("src/" + archivoXML);
DocumentBuilderFactory builderFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = builderFactory.newDocumentBuilder();
Document xmlDocument = builder.parse(XML);
XPath xPath = XPathFactory.newInstance().newXPath();
```

Tras esto, debemos conseguir un NodeList usando el DOM que conseguimos antes y usando como expresión "//persona" ya que es de ahí de dónde queremos sacar los datos.

```
NodeList nodelist = (NodeList)
xPath.compile("//persona").evaluate(xmlDocument,
XPathConstants.NODESET);
```

Debemos recorrer el NodeList para poder ir sacando los datos cogiendo cada nodo individualmente. Todos los datos se sacan de la siguiente manera:

```
Persona p = new Persona();
p.nombre = xPath.compile("@nombre").evaluate(nodo);
p.apellidos = xPath.compile("@apellidos").evaluate(nodo);
p.fechaNacimiento = xPath.compile("@fechaNacimiento").evaluate(nodo);
p.lugarNacimiento = xPath.compile("@lugarNacimiento").evaluate(nodo);
p.lugarResidencia = xPath.compile("@lugarResidencia").evaluate(nodo);
p.comentario = xPath.compile("@comentarios").evaluate(nodo);
```

Como las coordenadas, las fotos y los vídeos son elementos, debemos sacar sus nodos de otra manera:

```
Node nacimientocoord = (Node)
xPath.compile("coordenadasNacimiento").evaluate(nodo,
XPathConstants.NODE);
```

Y para conseguir los datos es de la misma forma que anteriormente, pero pasándole el nuevo nodo.

Lo siguiente es crear el árbol dándole valores a cada persona. Primero marcamos la altura que tiene cada persona en el árbol, siendo uno la primera (en mi caso la más a la izquierda) y teniendo altura 3 las últimas personas de la cadena, los amigos de los amigos de la que está en la altura 1. Lo segundo es marcar la posición que va en orden de 1 a 13. Por último, damos valor al booleano para marcar si la persona tiene amigos o no.

El siguiente paso es establecer la información de cada persona y cómo queremos que se vea, esto hay que hacerlo con código SVG. En mi caso, establecí que la información nacimiento de cada persona aparezca dentro de un rectángulo amarillo con borde morado. Los datos de altura y posición los usamos junto con nuevos datos de anchura, altura y demás para calcular el tamaño de los rectángulos y de la posición del texto dentro de ellos, es por eso por lo que pude colocar el lugar de nacimiento en otra línea distinta y de otro color. Con la etiqueta <rect> establecí las propiedades del rectángulo y con <text> la de la información.

A continuación, tuve que establecer las líneas que salen de cada rectángulo y hacia dónde se dirigían en el caso de que la persona tuviera amigos, ya que si no, no tiene caminos. Para ello usé la etiqueta <path> con la que puedo establecer el color, el tipo de línea y las coordenadas de salida y llegada (entre otras cosas) las cuales calculé con los datos de altura, posición y relleno, es decir, dependiendo de la altura y posición establecí unos valores u otros.

```
<path d="M 185 17 L 235 275 m 0 0 z" style="stroke:black"/>
```

Con “M x y” establezco las coordenadas absolutas de salida, con “L x y” hago que se forme una línea recta hacia esas coordenadas y con “m x y z”, como son letras minúsculas, significa que x e y son coordenadas relativas, por lo que marco que el final de la línea sea en el punto en el que estaba.

Lo último que queda es llamar al método *writelines* pasando como parámetros el archivo SVG dónde queremos escribir la lista de cadenas que también pasamos como parámetro.