```
/*
 * @(#)Stylizer.java 1.9 98/11/10
 *
 * Copyright (c) 1998 Sun Microsystems, Inc. All Rights Reserved.
 *
 * Sun grants you ("Licensee") a non-exclusive, royalty free, license to use,
 * modify and redistribute this software in source and binary code form,
 * provided that i) this copyright notice and license appear on all copies of
 * the software; and ii) Licensee does not utilize the software in a manner
 * which is disparaging to Sun.
 *
 * This software is provided "AS IS," without a warranty of any kind. ALL
 * EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY
 * IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR
 * NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BE
 * LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING
 * OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS
 * LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT,
 * INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER
 * CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF
 * OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGES.
 *
 * This software is not designed or intended for use in on-line control of
 * aircraft, air traffic, aircraft navigation or aircraft communications; or in
 * the design, construction, operation or maintenance of any nuclear
 * facility. Licensee represents and warrants that it will not use or
 * redistribute the Software for such purposes.
 */

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.FactoryConfigurationError;
import javax.xml.parsers.ParserConfigurationException;

import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;
import org.w3c.dom.Document;
import org.w3c.dom.DOMException;

// For write operation
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamSource;
import javax.xml.transform.stream.StreamResult;

import java.io.*;

public class Stylizer
{
    // Global value so it can be ref'd by the tree-adapter
    static Document document;

    public static void main (String argv [])
    {
        if (argv.length != 3) {
            System.err.println ("Usage: java Stylizer stylesheet xmlfile outfile");
            System.exit (1);
        }

        DocumentBuilderFactory factory =
            DocumentBuilderFactory.newInstance();
        //factory.setNamespaceAware(true);
        //factory.setValidating(true);

        try {
            File stylesheet = new File(argv[0]);
            File datafile   = new File(argv[1]);
            File outfile = new File(argv[2]);
            PrintStream out = new PrintStream(outfile);
            DocumentBuilder builder = factory.newDocumentBuilder();
            document = builder.parse(datafile);

            // Use a Transformer for output
            TransformerFactory tFactory =
                TransformerFactory.newInstance();
            StreamSource stylesource = new StreamSource(stylesheet);
            Transformer transformer = tFactory.newTransformer(stylesource);

            DOMSource source = new DOMSource(document);
            StreamResult result = new StreamResult(out);
            transformer.transform(source, result);

        } catch (TransformerConfigurationException tce) {
            // Error generated by the parser
            System.out.println ("\n** Transformer Factory error");
            System.out.println(" " + tce.getMessage() );

            // Use the contained exception, if any
            Throwable x = tce;
            if (tce.getException() != null)
                x = tce.getException();
            x.printStackTrace();

        } catch (TransformerException te) {
            // Error generated by the parser
            System.out.println ("\n** Transformation error");
            System.out.println(" " + te.getMessage() );

            // Use the contained exception, if any
            Throwable x = te;
            if (te.getException() != null)
                x = te.getException();
            x.printStackTrace();

         } catch (SAXException sxe) {
            // Error generated by this application
            // (or a parser-initialization error)
            Exception  x = sxe;
            if (sxe.getException() != null)
                x = sxe.getException();
            x.printStackTrace();

        } catch (ParserConfigurationException pce) {
            // Parser with specified options can't be built
            pce.printStackTrace();

        } catch (IOException ioe) {
            // I/O error
            ioe.printStackTrace();
        }

    } // main

}
```