Convertire un file CSV in XML con Java

La conversione di un file CSV (Comma-Separated Values) in XML è una pratica comune quando si ha bisogno di strutturare i dati in un formato leggibile e utilizzabile in applicazioni che richiedono un markup ben definito. In questo articolo, esploreremo come effettuare questa conversione utilizzando il linguaggio di programmazione Java.

Supponiamo di avere un file CSV chiamato `data.csv` con il seguente contenuto:

```
Name,Age,Email
John,30,john@example.com
Jane,25,jane@example.com
```

Quindi creiamo il codice principale:

```
import org.w3c.dom.Document;
import org.w3c.dom.Element;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import
javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
```

```
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class CsvToXml {
    public static void main(String[] args) {
        String csvFile = "data.csv";
        String line;
        String csvSplitBy = ",";
        try {
            // Creazione del documento XML
            DocumentBuilderFactory docFactory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder docBuilder =
docFactory.newDocumentBuilder();
            Document doc = docBuilder.newDocument();

            // Elemento radice
            Element rootElement =
doc.createElement("Rows");
            doc.appendChild(rootElement);

            // Lettura del CSV
            try (BufferedReader br = new
BufferedReader(new FileReader(csvFile))) {
                String[] headers =
br.readLine().split(csvSplitBy); // Leggi
l'intestazione
                while ((line = br.readLine()) !=
null) {
                    String[] data =
```

```
line.split(csvSplitBy);

                    // Crea un nuovo elemento per
ogni riga
                    Element row =
doc.createElement("Row");
                    rootElement.appendChild(row);

                    for (int i = 0; i <
headers.length; i++) {
                        Element element =
doc.createElement(headers[i]);

element.appendChild(doc.createTextNode(data[i]));
                        row.appendChild(element);
                    }
                }
            }

            // Scrivi il file XML su disco
            TransformerFactory transformerFactory =
TransformerFactory.newInstance();
            Transformer transformer =
transformerFactory.newTransformer();

transformer.setOutputProperty(OutputKeys.INDENT,
"yes");
            DOMSource source = new DOMSource(doc);
            StreamResult result = new
StreamResult(new FileWriter("data.xml"));

            transformer.transform(source, result);

            System.out.println("Conversione
```

```
completata! File XML creato: data.xml");

        } catch (IOException |
ParserConfigurationException |
javax.xml.transform.TransformerException e) {
            e.printStackTrace();
        }
    }
}
```

Ecco come appare il file `data.xml`:

```
<Rows>
    <Row>
        <Name>John</Name>
        <Age>30</Age>
        <Email>john@example.com</Email>
    </Row>
    <Row>
        <Name>Jane</Name>
        <Age>25</Age>
        <Email>jane@example.com</Email>
    </Row>
</Rows>
```

Conclusione

In questo articolo, abbiamo visto come convertire un file CSV in un file XML utilizzando Java. Questo approccio è altamente personalizzabile e può essere adattato a vari formati e requisiti. Se hai bisogno di una gestione più

complessa, puoi utilizzare librerie aggiuntive come OpenCSV o Jackson per la manipolazione dei dati CSV e XML.