Частное учреждение образования

«Колледж бизнеса и права»

Тема: «Работа с XML на языке Java»

Цель: Научиться создавать разметку приложения на языке Java средствами

XML.

Учащийся М.Н. Гончаров.

группа: Т-992

2022

6. Электроинструменты.

Электроинструменты можно структурировать по следующей схеме: – Model

– название модели; – Handy – одно- или двуручное; – Origin – страна

производства; – TC (должно быть несколько) – технические характеристики:

энергопотребление (низкое, среднее, высокое), производительность (в

единицах в час), возможность автономного функционирования и т.д.; –

Material – материал изготовления. Корневой элемент назвать PowerTools или

Power. С помощью XSL преобразовать XML-файл в формат XML, при

выводе корневым элементом сделать страну производства.

// main

import java.io.FileWriter;  
import java.io.IOException;  
import java.text.Format;  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.parsers.ParserConfigurationException;  
import javax.xml.transform.Transformer;  
import javax.xml.transform.TransformerConfigurationException;  
import javax.xml.transform.TransformerException;  
import javax.xml.transform.TransformerFactory;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import javax.xml.transform.stream.StreamSource;  
  
/\*  
  
 Инструменты  
 <PowerTools>  
 <model> Name </model>  
 <Handy> одно- или двуручное </Handy>  
 <Origin> Страна Производитель</Origin>  
 <TS>  
 <UseEnergy> (низкое, среднее, высокое </UseEnergy>  
 <Perfomance> Ten </Perfomance>  
 </TS>  
 <material> </material>  
 </PowerTools>  
  
\* \*/  
  
public class laba\_17 {  
 public static void main(String[] args) {  
 // Create class /// String model, String handy, String origin, String UseEnergy, int Perfomance  
 PowerTools ps = new PowerTools("Svarka", "one\_handy", "Belarus", "Medium", "Ten");  
  
 /// Как Улучшение можно сделать Класс и метод который будет выбирать кол-во Node и их содержимое  
 /\* START ======================================================================================== Create And append Data in \*.xml \*/  
 DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder documentBuilder = null;  
 try  
 {  
 documentBuilder = documentBuilderFactory.newDocumentBuilder();  
 }  
 catch (ParserConfigurationException e)  
 {  
 e.printStackTrace();  
 } // forming a document tree  
  
 // Document  
 Document document = documentBuilder.newDocument();  
  
 // root  
 String root = "PowerTools";  
 Element rootElement = document.createElement(root);  
 document.appendChild(rootElement);  
  
 // model  
 Element elementName = document.createElement("model");  
 String name = ps.getModel();  
 elementName.appendChild(document.createTextNode(name));  
  
 // handy  
 Element elementHandy = document.createElement("handy");  
 String handy = ps.getHandy();  
 elementHandy.appendChild(document.createTextNode(handy));  
 //elementHandy.setAttribute("id", "777");  
  
 // Origin  
 Element elementOrigin = document.createElement("origin");  
 String Origin = ps.getOrigin();  
 elementOrigin.appendChild(document.createTextNode(Origin));  
  
 // TS  
 Element elementTS = document.createElement("ts");  
 // child elements TS - UseAnergy, Perfomance  
 Element elementUseAnergy = document.createElement("UseAnergy");  
 String UseEnergy = ps.getUseEnergy();  
 elementUseAnergy.appendChild(document.createTextNode(UseEnergy));  
 Element elementPerfomance = document.createElement("Perfomance");  
 String Perfomance = ps.getPerfomance();  
 elementPerfomance.appendChild(document.createTextNode(Perfomance));  
  
 // append child TS  
 elementTS.appendChild(elementUseAnergy);  
 elementTS.appendChild(elementPerfomance);  
  
 // Material  
 Element elementMaterial = document.createElement("material");  
 String material = "Metal";  
 elementMaterial.appendChild(document.createTextNode(material));  
  
 // write tree to file  
 rootElement.appendChild(elementName);  
 rootElement.appendChild(elementHandy);  
 rootElement.appendChild(elementOrigin);  
 rootElement.appendChild(elementTS);  
 rootElement.appendChild(elementMaterial);  
  
 TransformerFactory transformerFactory = TransformerFactory.*newInstance*();  
 try  
 {  
 Transformer transformer = transformerFactory.newTransformer();  
 DOMSource source = new DOMSource(document);  
 StreamResult result = new StreamResult(new FileWriter("PowerTools.xml"));  
 transformer.transform(source, result);  
 }  
 catch (TransformerConfigurationException e)  
 {  
 e.printStackTrace();  
 }  
 catch (TransformerException | IOException e)  
 {  
 e.printStackTrace();  
 }  
 /\* END ======================================================================================== Create And append Data in \*.xml \*/  
  
 try  
 {  
 TransformerFactory tFact = TransformerFactory.*newInstance*();  
 Transformer transformer = tFact.newTransformer(new StreamSource("PowerTools.xsl"));  
 transformer.transform(new StreamSource("PowerTools.xml"), new StreamResult("PowerTools.html"));  
 }  
 catch (TransformerException e)  
 {  
 e.printStackTrace();  
 }  
 }  
}

// class powertools

/\*  
  
 Инструменты  
 <PowerTools>  
 <model> Name </model>  
 <Handy> одно- или двуручное </Handy>  
 <Origin> Страна Производитель</Origin>  
 <TS>  
 <UseEnergy> (низкое, среднее, высокое </UseEnergy>  
 <Perfomance> Ten </Perfomance>  
 </TS>  
 <material> </material>  
 </PowerTools>  
  
\* \*/  
  
public class PowerTools {  
 public String getModel() {  
 return model;  
 }  
  
 public void setModel(String model) {  
 this.model = model;  
 }  
  
 public String getHandy() {  
 return handy;  
 }  
  
 public void setHandy(String handy) {  
 this.handy = handy;  
 }  
  
 public String getOrigin() {  
 return origin;  
 }  
  
 public void setOrigin(String origin) {  
 this.origin = origin;  
 }  
  
 public String getUseEnergy() {  
 return UseEnergy;  
 }  
  
 public void setUseEnergy(String useEnergy) {  
 UseEnergy = useEnergy;  
 }  
  
 public String getPerfomance() {  
 return Perfomance;  
 }  
  
 public void setPerfomance(String perfomance) {  
 Perfomance = perfomance;  
 }  
  
 private String model;  
 private String handy;  
 private String origin;  
 private String UseEnergy;  
 private String Perfomance;  
  
 public PowerTools() {  
 }  
  
 public PowerTools(String model, String handy, String origin, String UseEnergy, String Perfomance) {  
 this.model = model;  
 this.handy = handy;  
 this.origin = origin;  
 this.UseEnergy = UseEnergy;  
 this.Perfomance = Perfomance;  
 }  
}

// powertools.xsl

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">  
 <xsl:template match="/">  
 <html>  
 <head>  
 <title>PowerTools</title>  
 </head>  
 <body><table border="1">  
 <tr>  
 <th>model</th>  
 <th>handy</th>  
 <th>origin</th>  
 <tr>  
 <th>UseAnergy</th>  
 <th>Perfomance</th>  
 </tr>  
 <th>material</th>  
 </tr>  
 <xsl:for-each select="PowerTools/PowerTools">  
 <tr>  
 <td><xsl:value-of select="@model"/></td>  
 <td><xsl:value-of select="handy"/></td>  
 <td><xsl:value-of select="origin"/></td>  
 <td><xsl:value-of select="ts/UseAnergy"/></td>  
 <td><xsl:value-of select="ts/Perfomance"/></td>  
 <td><xsl:value-of select="material"/></td>  
 </tr>  
 </xsl:for-each>  
 </table>  
 </body>  
 </html>  
 </xsl:template>  
 </xsl:stylesheet>

// rezult

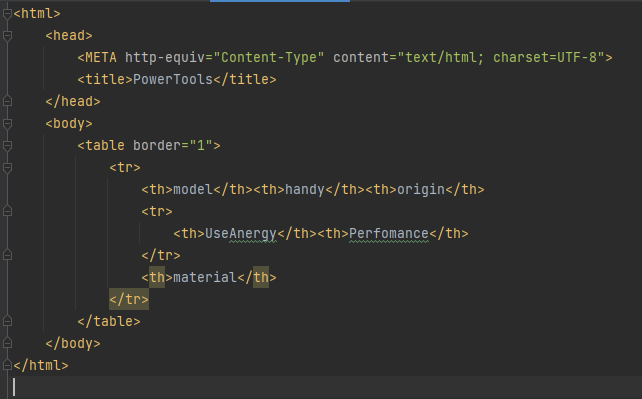


Рисунок 1

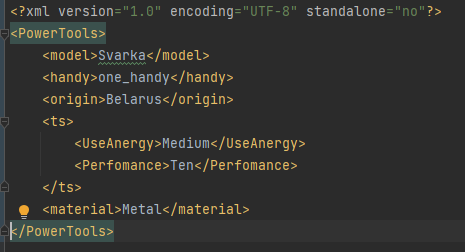


Рисунок 2