分散システム

第八回レポート課題

課題 1

【課題内容】

これまでの課題で扱ってきた商品データ XML 文書を、アプリケーションデータとして格納するための Java クラスを定義してみよ。また、XML 文書の各要素と各 Java クラスのメンバとの関連や、XML 文書を Java クラスに読み込む方法について説明を加えよ。

余力があれば、XML 文書を上で定義した Java クラスのインスタンスとして読み込むプログラムを作成してみよ。

【作成したソースコード】

≪Itemlist.java≫

```
/**
 * XML 文書の<itemlist>タグに対応する Java の Itemlist クラス
 */

import java.util.*;

import javax.xml.parsers.*;
import org.w3c.dom.*;

public class Itemlist {
    private Vector<Item> itms;

    public Itemlist() {
        this.itms = new Vector<Item>();
    }

    public Vector<Item> getEmps() {return itms;}
    public void setEmps(Vector<Item> v) {
        this.itms = v;
    }

    public void addItem(Item itm) {
        itms.add(itm);
    }

    /**
     * Itemlist オブジェクトの文字列表現を生成する
     */
    public String toString() {
        StringBuffer str = new StringBuffer();
        str.append("--- Items ---¥n");
        for(int i=0;i<itms.size();i++) {
            str.append(itms.get(i).toString());
        }
        return str.toString();
    }

    /**
```

```
     * XML 文書の itemlist 要素から Itemlist オブジェクトを構成する
     * @param e itemlist 要素
     * @return Itemlist オブジェクト
     */
    public static Itemlist unmarshall(Element e) {
        Itemlist iteml = new Itemlist();
        // すべての子ノードを調べる
        for(Node c1 = e.getFirstChild();c1 != null; c1 = c1.getNextSibling()) {
            if(c1 instanceof Element) {
                if(c1.getNodeName().equals("item")) { // item 要素の時
                    Item itm = Item.unmarshall((Element)c1); // Item オブジェクトの作成
                    iteml.itms.add(itm);   // Item オブジェクトを追加する
                }
            }
        }
        return iteml;
    }

    public static void main(String[] args) {
        try {
            // DOM パーサーの生成
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            // XML 文書の解析
            Document doc = builder.parse(args[0]);
            // XML 文書から Itemlist オブジェクトへ unmashall する
            Itemlist iteml = Itemlist.unmarshall(doc.getDocumentElement());
            System.out.println("current time = " + new
java.util.Date(System.currentTimeMillis()));
            // Itemlist オブジェクトの表示
            System.out.println(iteml.toString());
        } catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

≪item.java≫

```
/**
 * XML 文書の<item>タグに対応する Java の Item クラス
 */

import org.w3c.dom.*;

public class Item {
    private String name;
    private String value;

    public Item() {
        this.name = "";
        this.value = "";
    }

    public void setName(String name) {
        this.name = name;
    }
    public String getName() {return name;}

    public void setValue(String value) {
        this.value = value;
    }
    public String getValue() {return value;}
```

```
    /**
     * Item オブジェクトの文字列表現を生成する
     */
    public String toString() {
        return "Item name = " + name + ", value = " + getValue() + "¥n";
    }

    /**
     * XML 文書の item 要素から Item オブジェクトを構成する
     * @param e item 要素
     * @return Item オブジェクト
     */
    public static Item unmarshall(Element e) {
        Item itm = new Item();
        // すべての子ノードを調べる
        for(Node c1=e.getFirstChild();c1!=null;c1=c1.getNextSibling()) {
            if(c1 instanceof Element) {
                if(c1.getNodeName().equals("name")) {    // name 要素の時
                    itm.setName(c1.getTextContent());
                } else if(c1.getNodeName().equals("value")) {    // value 要素の時
                    itm.setValue(c1.getTextContent());
                }
            }
        }
        return itm;
    }
}
```

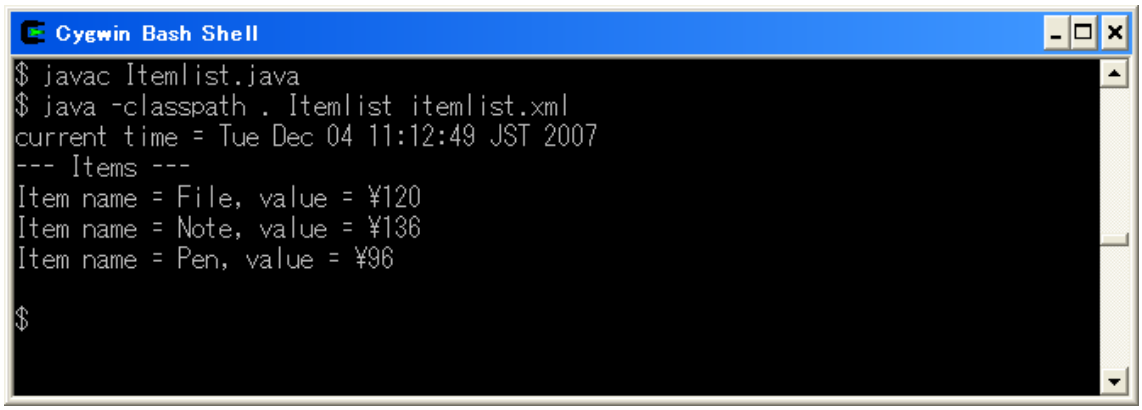
【読み込ませた XML 文書ファイル】

≪itemlist.xml≫

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE ItemList SYSTEM "itemlist.dtd">

<itemlist>
    <item>
        <name>File</name>
        <value>¥120</value>
    </item>
    <item>
        <name>Note</name>
        <value>¥136</value>
    </item>
    <item>
        <name>Pen</name>
        <value>¥96</value>
    </item>
</itemlist>
```

【実行結果】



【考察】

XML 文書の各要素と各 Java クラスのメンバとの関連

Itemlist クラスは、「商品リスト」に対応するクラスである。このクラスには、Item 型の配列(Vector) itms メンバがある。このメンバは、XML 文書内の複数ある item 要素のデータを格納する配列の役割を担っている。

Item クラスは、「商品」に対応するクラスである。このクラスは、String 型の name メンバと value メンバを持つ。name メンバは商品名を、value メンバは価格をそれぞれ格納する役割を担っている。

XML 文書を Java クラスに読み込む方法

このプログラムは、まず XML 文書の DocumentElement を引数とし、Itemlist クラスの unmarhsall メソッドを呼び出す。Itemlist クラスの unmarshall メソッドは、受け取った要素のすべての子ノードを調べる(FirstChild→NextSibling の流れにより)。そして、見つかったら文字列"item"と比較し、一致したら Item クラスの unmarshall メソッドを呼び出す。

Item クラスの unmarshall メソッドは Itemlist クラスと同様、受け取った要素のすべての子ノードを調べる。そして、見つかった場合は、文字列"name"と一致したら name 要素の値を name メンバに読み込み、文字列"value"と一致したら value 要素の値を value メンバに読み込む。

以上のような一連の操作により、XML 文書を Java クラスに読み込んでいる。