u第7,8,9章●MVCアーキテクチャの実装(練習問題)

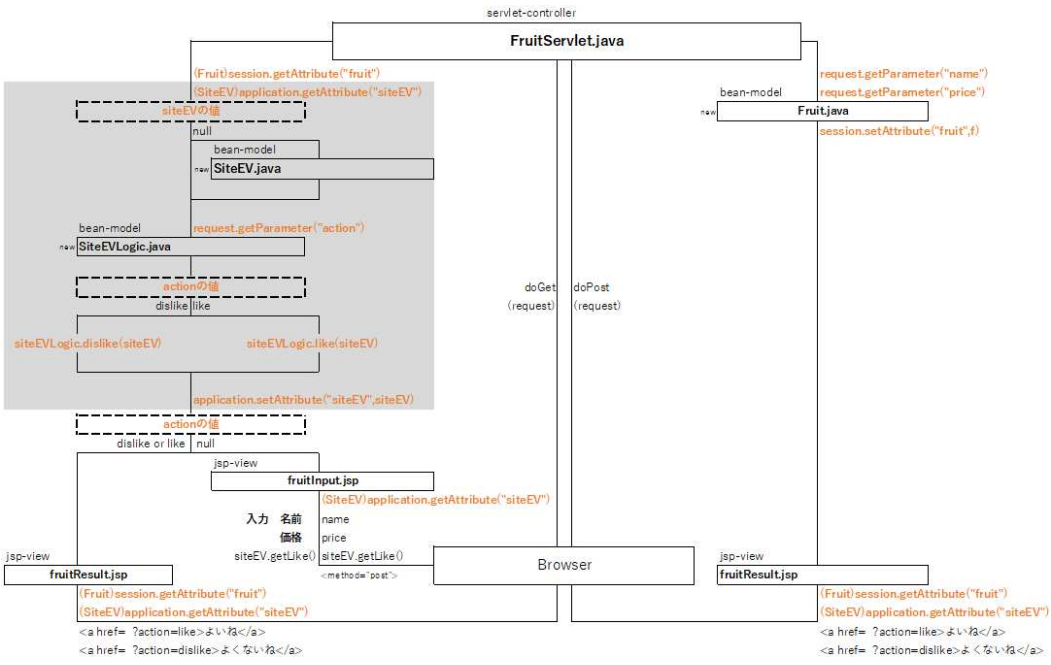
フルーツの税込み価格計算Webアプリケーションを MVCモデルで作成（評価リンク付き）

練習問題の内容をベースとした「フルーツの税込み価格計算」Webアプリケーションを作成する。評価リンクを実装するなど、足りない仕様を加えた内部設計は以下のとおりとする。

【仕様】

・入力画面を作成し果物の名前と価格を入力できるようにする
・入力画面には計算ボタンを配置しクリックすることにより税込み価格画面に遷移する
・入力画面には「よいね・よくないね」の評価状況（カウント数）を表示する
・税込み価格画面では果物名と10%の消費税込みの金額を表示する
・税込み価格画面には「よいね・よくないね」の評価リンクを設置する
・評価リンクのクリックで即座にカウント数がアップされる
・カウント数は価格入力画面でも税込み価格画面でも確認可能とする

【内部設計】



【実装】

①Fruit.javaクラスを作成する

練習７－２のコードを、10%の消費税を上乗せした価格を保持するように変更する。

パッケージ名は jp.ict.aso.model とする。

```
1  package jp.ict.aso.model;
2
3  import java.io.Serializable;
4
5  public class Fruit implements Serializable {
6    private double price;
7    private String name;
8    public Fruit(){
9    }
10    public Fruit(String name , double price){
11      this.name=name;
12      this.price=price;
13    }
14    public double getPrice() {
15      return price*1.1;
16    }
17    public String getName() {
18      return name;
19    }
20  }
```

②FruitServlet.javaクラスを作成する

doGet()メソッドではサイト評価処理を行い、データ入力か結果出力かのフォワード先を actionパラメータで判断する。

doPost()メソッドではリクエストパラメータを取得しBeanに設定・実体化し、セッションスコープに保存して結果出力へフォワードする。

アノテーション（urlパラメータ）は /Fruit とする。

パッケージ名は jp.ict.aso.controller とする。

HealthCheck.javaやMinatoIndex.javaの必要部分をコピーして効率よく実装します。（黄色い部分はコード9-5（MinatoIndex.java）の内容をほぼコピーしています）

```
1  package jp.ict.aso.controller;
2
3  import java.io.IOException;
4  import javax.servlet.RequestDispatcher;
```

```
 5  import javax.servlet.ServletException;
 6  import javax.servlet.ServletContext;
 7  import javax.servlet.annotation.WebServlet;
 8  import javax.servlet.http.HttpServlet;
 9  import javax.servlet.http.HttpServletRequest;
10  import javax.servlet.http.HttpServletResponse;
11  import javax.servlet.http.HttpSession;
12  import model.*;
13  import jp.ict.aso.model.*;
14
15  @WebServlet("/Fruit")
16  public class FruitServlet extends HttpServlet {
17
18    protected void doGet(HttpServletRequest request,
19        HttpServletResponse response)
20        throws ServletException, IOException {
21      // セッションスコープに保存された登録ユーザ
22      HttpSession session = request.getSession();
23      Fruit fs = (Fruit) session.getAttribute("fruit");
24  ////////////////////////////////////////////////////////////
25      // アプリケーションスコープに保存されたサイト評価を取
得
26      ServletContext application = this.getServletContext();
27      SiteEV siteEV = (SiteEV)
application.getAttribute("siteEV");
28
29      // サイト評価の初期化（初回リクエスト時実行）
30      if (siteEV == null) {
31        siteEV = new SiteEV();
32      }
33
34      // リクエストパラメータの取得
35      request.setCharacterEncoding("UTF-8");
36      String action = request.getParameter("action");
37
38      if( action != null && fs != null){
39      // サイトの評価処理（初回リクエスト時は実行しない）
40        SiteEVLogic siteEVLogic = new SiteEVLogic();
41        if (action != null && action.equals("like")) {
42          siteEVLogic.like(siteEV);
43        } else if (action != null && action.equals("dislike")) {
44          siteEVLogic.dislike(siteEV);
45        }
```

```
46      }
47      // アプリケーションスコープにサイト評価を保存
48      application.setAttribute("siteEV", siteEV);
49  ////////////////////////////////////////////////////////////
50      // フォワード
51      if(action == null){
52        RequestDispatcher dispatcher =
53          request.getRequestDispatcher
54              ("/WEB-INF/jsp/fruitInput.jsp");
55        dispatcher.forward(request, response);
56      }else{
57        RequestDispatcher dispatcher =
58          request.getRequestDispatcher
59              ("/WEB-INF/jsp/fruitResult.jsp");
60        dispatcher.forward(request, response);
61      }
62    }
63    protected void doPost(HttpServletRequest request,
64        HttpServletResponse response)
65        throws ServletException, IOException {
66
67      // リクエストパラメータを取得
68      request.setCharacterEncoding("UTF-8");
69      String name = request.getParameter("name"); // 名前
70      String strprice = request.getParameter("price"); // 価
格
71      double price = Double.parseDouble(strprice);
72
73      // 入力値をプロパティに設定
74      Fruit f = new Fruit(name,price);
75
76      // セッションスコープに保存
77      HttpSession session = request.getSession();
78      session.setAttribute("fruit",f);
79
80      // フォワード
81      RequestDispatcher dispatcher =
82          request.getRequestDispatcher
83              ("/WEB-INF/jsp/fruitResult.jsp");
84      dispatcher.forward(request, response);
85    }
86  }
87
```

③fruitInput.jsp を作成する

果物の名前と価格を入力しpostメソッドで urlパターンのFruit へリクエストする。 評価状況（カウント数）も表示する。

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>

<%@ page import="model.SiteEV" %>
<%
// アプリケーションスコープに保存されたSiteEVを取得
SiteEV siteEV = (SiteEV) application.getAttribute("siteEV");
%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>フルーツ</title>
</head>
<body>
<h1>フルーツ税込み価格計算</h1>
<form action="Fruit" method="post">
名前：<input type="text" name="name">
価格：<input type="text" name="price">
<input type="submit" value="税込み価格">
</form>

<p>
よいね：
<%= siteEV.getLike() %>人
よくないね：
<%= siteEV.getDislike() %>人
</p>

</body>
</html>

```

④fruitResult.jsp を作成する

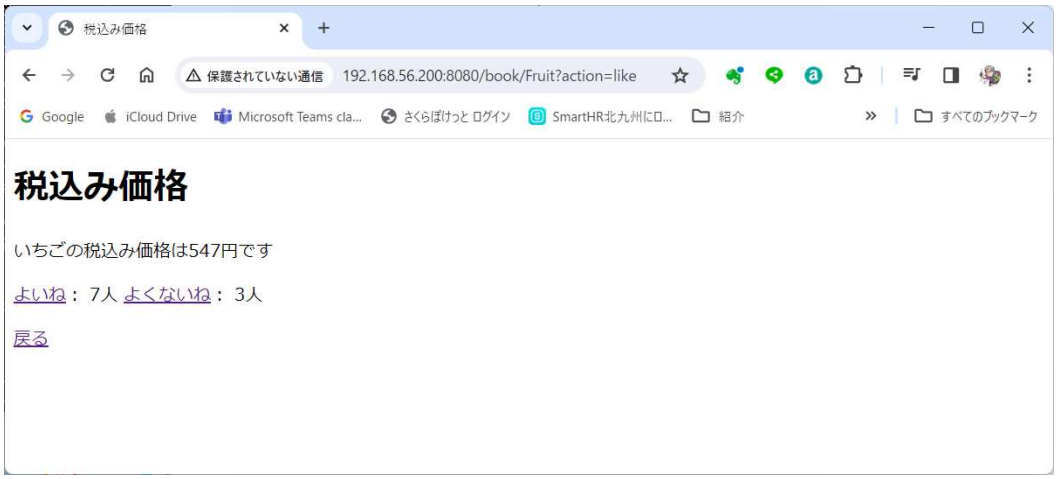
果物の名前と税込み価格と評価リンクを出力する。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ page import="jp.ict.aso.model.Fruit" %>
<%@ page import="model.SiteEV" %>
<%
// セッションスコープに保存されたfruitを取得
Fruit fs = (Fruit)session.getAttribute("fruit");
// アプリケーションスコープに保存されたsiteEVを取得
SiteEV siteEV = (SiteEV) application.getAttribute("siteEV");
%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>税込み価格</title>
</head>
<body>
<h1>税込み価格</h1>
<%= fs.getName() %>の税込み価格は<%= (int)fs.getPrice() %>円です
<p>
<a href="Fruit?action=like">よいね</a> :
<%= siteEV.getLike() %>人
<a href="Fruit?action=dislike">よくないね</a> :
<%= siteEV.getDislike() %>人
</p>
<a href="Fruit">戻る</a></body>

</body>
</html>

```

※完成したら**実行画面を提出**してください

【実行例】

↓urlパターンが見えていること



【解説】

ポイントは、**SiteEVクラスはアプリケーションスコープ、Fruitクラスはセッションスコープ**で実装することです。

ServletクラスのdoGet()メソッド部分でイイネのSiteEVのインスタンスを受け渡すときはapplicationスコープでないといけません。これは疑似的なグローバル変数としてSiteEVを設定したいわけですから、リクエスト変数やセッション変数は使用できません。
また、結果出力のjspがSiteEVのインスタンスを受け取るときも当然アプリケーションスコープでgetすることになります。

一方でFruitクラスのBeanをServlet側でリクエストスコープで渡すと、リクエストをまたいでインスタンスを送ることができなくなります（リクエストスコープは1回しか渡せません）。つまり「よいね・よくないね」リンクを押した段階でaction指定とともにServletへGETリクエストでページ遷移が発生することになるわけですから**Fruitのインスタンスはなくなってしまいます**。つまりServletのdoGet()メソッドでSiteEVが処理されて、リンクを踏んだ結果出力用jspに戻ってきた段階では複数ページをまたいでいるので

NullPointerExceptionが発生してしまうということです。これは結果出力後30分間放置してセッションタイムアウトが起こった時にも発生してしまいます。

よってFruitクラスのインスタンスはページをまたげるセッションスコープかアプリケーションスコープでしか渡せません。
（教科書p263に表になって出ています）
結果出力のjsp側も、当然Servletクラスから送られてきたスコープに合わせてインスタンスを受け取ることになります。

一方で、アプリケーションスコープでFruitクラスのインスタンスを渡すとグローバル変数扱いになって、入力した名前と価格が他のブラウザからも見えてしまいます。よって、ここはセッションリクエストしか指定できないということになります。