



# コンビニエンスストア店舗検索システム - JavaEE8, PostgreSQL



office・M

2024年9月7日 14:50

...

JavaEE8 (JSP・Servlet) の環境でWebアプリケーションの開発方法を学ぶ講座をシリーズで提供しています。今回は基本情報技術者試験のソフトウェア設計問題として過去出題された仕様をもとにプログラムを設計し実装します。

とりわけ、DBを操作するクラスファイルを実装する方法を学びます (DAOを作成します)。チーム開発での役割分担を想定しています。

2024年9月よりECLIPSEのバージョンを最新版 (Version: 2024-06 (4.32.0)) に変更しました。

## ▼ 目次

開発概要

開発方針

DB設計

論理設計

物理設計

外部設計

内部設計

SunkusHtmlBean.classの仕様とクラス図

---

SunkusUpdateHtmlBean.classの仕様とクラス図

---

## 実装準備

---

PostgreSQLのjdbcドライバを配置

---

## 実装手順

---

1. DAO用JavaBeansクラスを作成します
  2. リクエストコントロール用Servletクラスを作成します
  2. お気に入り店舗検索用画面のJSPファイルを作成します
  3. お気に入り数更新用画面のJSPファイルを作成します
- 

## 実行確認

---

1. 正常に実行されることを確認します
  2. スタイルシートや背景画像で簡単に装飾します
  3. 装飾を確認します
- 

## 仕様反映

---

1. SunkusDropDownListBean.classの設計を追加します
  2. SunkusHtmlButtonBean.classの設計を追加します
  3. 追加設計したJavaBeansクラスを実装します
  4. リクエストコントロール用Servletクラスを修正します
  5. お気に入り店舗検索用画面のJSPファイルを修正します
  6. お気に入り数更新用画面のJSPファイルを修正します
- 

## 最終確認

# 開発概要

JavaEE8アプリケーション・サーバ（Tomcat）とデータベース・サーバ（PostgreSQL）を利用して開発を行うということを前提に実装を進めていきます。開発ツールには統合開発環境のEclipseを用いることにします。

**題材として『基本情報技術者試験の平成28年春期の午後問5（ソフトウェア設計）』を取り上げます。具体的にはコンビニエンスストアの店舗検索システムをWebアプリケーションのMVCモデルで作成します（試験問題の中ではWeb方式と呼んでいます）。**

JavaEE8環境ですので使用言語は当然Java（JSP・Servlet）となり、設計技法にはMVCモデルを使います。MVCモデルとはWebアプリケーションの構成をModel（業務ロジック）－View（表示）－Controller（制御）に分割して設計する技法です。3つのモデルに役割分担することで部品化が促され、ひいてはチーム開発に貢献します。

**今回は、データベースの接続や抽出処理など業務ロジックを実現するBeanも作成します。**

実装していく過程でEclipseの利用方法が不明な場合は、別途メール等でお知らせください。この講座では、作成済みの仕様と設計書から「**JavaEE8環境でのMVCモデルの実装の仕組みを学ぶ**」ことを目的とします。

## 開発方針

Webアプリケーションの開発環境には図1のようにEclipse2024を用います。



図1. 開発環境 Eclipse2024

Spring等のフレームワークは使用しません。これは、MVCモデルにおけるhttpプロトコルの処理の実装を直に学んでいただきたいためです。

開発に必要な最小限の設計ドキュメントとソースコードの例はすべて提示します。この情報をもとに、自分で試行錯誤しながら実装してみてください。  
設計のもとになる仕様は基本情報技術者試験の問題文に提示されています。以下のサイトから参照して、内容をしっかりと把握してください。

**プログラムは問題文において提示されたWeb方式で開発することにします。**

# 基本情報技術者過去問題 平成28年春期 午後問5(ソフトウェア設計)

www.fe-siken.com

## DB設計

### 論理設計

基本情報技術者試験の問題文通りに設計します。

|      |     |       |    |      |      |
|------|-----|-------|----|------|------|
| 店舗番号 | 店舗名 | 市区町村名 | 住所 | 電話番号 | 特売情報 |
|------|-----|-------|----|------|------|

図2. 店舗データテーブル (T\_店舗) のレコード形式

|      |        |
|------|--------|
| 店舗番号 | お気に入り数 |
|------|--------|

図3. お気に入りデータテーブル (T\_お気に入り) のレコード形式

### 物理設計

接続するPostgreSQLサーバのDB名は「**mrp**」とします。

#### (DB構造,スキーマ)

```
create table T_店舗(  
  店舗番号 int not null,  
  店舗名 varchar(30),  
  市区町村名 varchar(30),  
  住所 varchar(30),  
  電話番号 varchar(20),  
  特売情報 varchar(20),  
  PRIMARY KEY (店舗番号));  
create table T_お気に入り(  
  店舗番号 int not null,  
  お気に入り数 int,  
  PRIMARY KEY (店舗番号));
```

**A5 : SQL Mk-2などのSQLクライアントソフトを使ってSQL文を流し込んでください。店舗データテーブル（T\_店舗）とお気に入りテーブル（T\_お気に入り）が作成されサンプルデータが入力されます。**



図4. V\_お気に入り店舗の出力

ファイル→新規→動的Webプロジェクト→「sunkusfe」プロジェクトを作成する→図5の内容で設定する



※作成済みであればこの処理は必要ありません

urlにアクセスすると図6のような、コンビニ店舗検索・更新画面が表示されます。市区町村名で対象店舗を絞り込み、店舗番号を選択するか直接店舗レコードの更新ボタンから、お気に入り数を更新します。スマートフォンでの運用が前提のため絞り込みや選択にはドロップダウンリストを用いレコード毎に更新ボタンを設置します。



図6. コンビニ店舗検索・更新画面

## 内部設計

クラス連携図は図7のようになります。

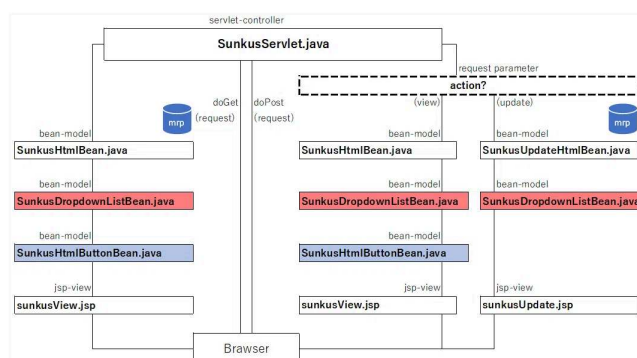


図7. MVCモデル図（クラス連携図）

図7のMVCモデル図において、実装を単純化するために**まずは白抜き枠の部分の仕様**を実現することにします。枠の背景が赤と青のクラスは、ドロップダウンリスト（選択）とレコード内での更新ボタンの設置の役割を担うクラスになります。これら選択と更新を行う機能は「単純化した実装」の動きを確認したうえで、改めて**仕様反映**として実現します。

白抜き枠のBeanの使い方とクラス図は以下のようにになります。

## SunkusHtmlBean.classの仕様とクラス図

(仕様)

- ・データベースから店舗情報を**検索し表示**を行います。
- ・コンストラクタの引数でPostgreSQLサーバのIPとDB名を指定して実体化します。
- ・setInputCity()メソッドに市区町村名をセットしてsunkusSearch()メソッドを実行することで対象となる地域の店舗情報がDBから検索されます。
- ・検索結果はhtml形式で表組整形されてresultSetフィールドに格納されます。
- ・利用側のクラスはgetResultSet()メソッドで検索結果を取得します。

(クラス図)

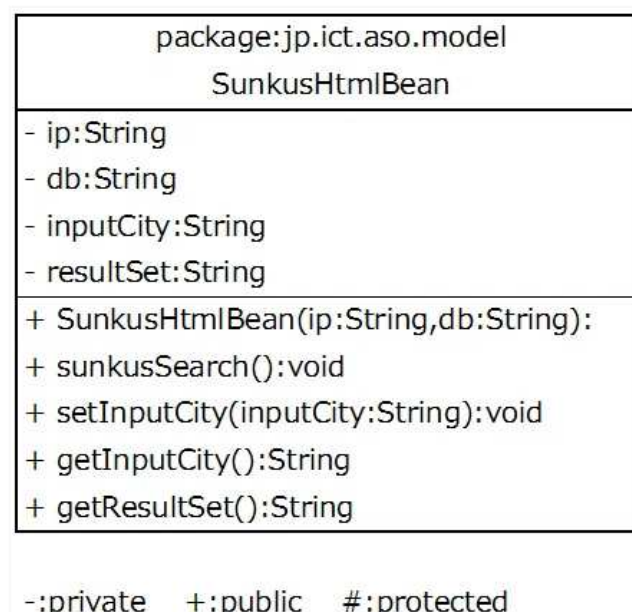


図8. SunkusHtmlBeanクラス図

## SunkusUpdateHtmlBean.classの仕様とクラス図

(仕様)

- ・データベースの店舗情報を**更新し表示**を行います。
- ・コンストラクタの引数でPostgreSQLサーバのIPとDB名を指定して実体化します。
- ・setInputNumber()メソッドに店舗番号をセットしてsunkusUpdate()メソッドを実行することで対象店舗の「お気に入り数」が+ 1 インクリメントされます。
- ・更新された店舗情報はhtml形式で表組整形されてresultSetフィールドに格納されます。
- ・利用側のクラスはgetResultSet()メソッドで更新店舗情報を取得します。

(クラス図)

|                                                                                                                                                                             |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jp.ict.aso.model<br>SunkusUpdateHtmlBean                                                                                                                                    |
| - ip : String<br>- db : String<br>- inputNumber : int<br>- resultSet : String                                                                                               |
| + SunkusUpdateHtmlBean(ip: String, db: String)<br>+ sunkusUpdate(): void<br>+ setInputNumber(inputNumber: int): void<br>+ getInputNumber(): int<br>+ getResultSet(): String |
| -:private   +:public   #:protected                                                                                                                                          |

図9. SunkusUpdateHtmlBeanクラス図

## 実装準備

### PostgreSQLのjdbcドライバを配置

**org.postgresql.driver.jar** ファイルをパッケージエクスプローラ内の**sunkusfe**プロジェクトの**src/main/webapp/WEB-INF/lib**フォルダにドラッグ&ドロップして登録します。

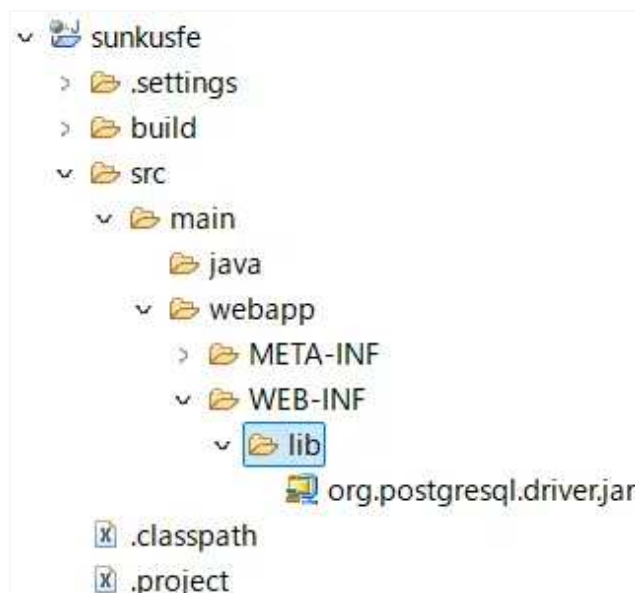


図10. PostgreSQLドライバの配置位置

## 実装手順

### 1. DAO用JavaBeansクラスを作成します



---

Eclipseパッケージ・エクスプローラより  
sunkusfeプロジェクトを右クリック→新規→クラス  
→以下の内容で作成する

---

- SunkusHtmlBean.java
  - パッケージ : jp.ict.aso.model
  - クラス名 : SunkusHtmlBean
  - ソースコード例 : 図8のクラス図より以下のように実装する

```
1 package jp.ict.aso.model;
2
3 import java.io.Serializable;
4 import java.sql.Connection;
5 import java.sql.DriverManager;
6 import java.sql.ResultSet;
7 import java.sql.Statement;
8
9 public class SunkusHtmlBean implements Serializable {
10     private String ip;
11     private String db;
12     private String inputCity;
13
14     private String resultSet;
15
16     public SunkusHtmlBean(String ip, String db) {
17         resultSet = "";
18         inputCity = "";
19         this.ip = ip;
20         this.db = db;
21     }
22
23     public void sunkusSearch() {
24         int number;
25         String name;
26         String city;
27         String address;
28         String tel;
29         String tokubai;
30         int okiniiri;
```

図11. SunkusHtmlBean.java (1)

```
31     try {
32         // [1] JDBCドライバのロード
33         Class.forName("org.postgresql.Driver");
34         // [2] RDBへの接続
35         String url = jdbc:postgresql://" + ip + ":5432/" + db;
36         Connection connection =
37             DriverManager.getConnection(url, "postgres", "postgres");
38         // [3] SQL文のコンパイル作成
39         Statement stmt = connection.createStatement();
40         // [4] SQL文を実行する
41         ResultSet rs = stmt.executeQuery(
42             "select * from v_お気に入り店舗 "
43             + "where 市区町村名 like '%" + inputCity + "%' order by 店舗番号");
44         // [5] RDBの検索結果を取り出す
45         resultSet = resultSet + "table border=1";
46         resultSet = resultSet + "<tr><th>店舗番号</th><th>店舗名</th><th>市町村区名</th>";
47         resultSet = resultSet + "<th>住所</th><th>電話番号</th><th>特売情報</th><th>お気に入り数</th></tr>";
48
49         while (rs.next()) {
50             number = rs.getInt("店舗番号");
51             name = rs.getString("店舗名");
52             city = rs.getString("市区町村名");
53             address = rs.getString("住所");
54             tel = rs.getString("電話番号");
55             tokubai = rs.getString("特売情報");
56             okiniiri = rs.getInt("お気に入り数");
57             resultSet = resultSet + "<tr><td>" + number + "</td><td>" + name + "</td><td>" + city + "</td><td>";
58             resultSet = resultSet + address + "</td><td>" + tel + "</td><td>" + tokubai + "</td><td>" + okiniiri + "</td></tr>";
59         }
60         resultSet = resultSet + "</table>";
```

図12. SunkusHtmlBean.java (2)

```
61
62 // [6] データベースから切断する
63 stmt.close();
64 connection.close();
65 } catch (Exception e) {
66     e.printStackTrace();
67 }
68
69
70 public void setInputCity(String inputCity) {
71     this.inputCity = inputCity;
72 }
73
74 public String getInputCity() {
75     return inputCity;
76 }
77
78 public String getResultSet() {
79     return resultSet;
80 }
```

図13. SunkusHtmlBean.java (3)

---

Eclipseパッケージ・エクスプローラより  
sunkusfeプロジェクトを右クリック→新規→クラス

→以下の内容で作成する

- SunkusUpdateHtmlBean.java

パッケージ : jp.ict.aso.model

クラス名 : SunkusUpdateHtmlBean

ソースコード例 : 図9のクラス図より以下のように実装する

```
1 package jp.ict.aso.model;
2 import java.io.Serializable;
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.Statement;
7
8 public class SunkusUpdateHtmlBean implements Serializable {
9     private String ip;
10    private String db;
11    private int inputNumber;
12
13    private String resultSet;
14
15    public SunkusUpdateHtmlBean(String ip, String db) {
16        resultSet = "";
17        this.ip = ip;
18        this.db = db;
19    }
20    public void sunkusUpdate() {
21        int number;
22        String name;
23        String city;
24        String address;
25        String tel;
26        String tokubai;
27        int okiniiri;
28        try {
29            // [1] JDBCドライバのロード
30            Class.forName("org.postgresql.Driver");
31            // [2] RDBへの接続
```

図14. SunkusUpdateHtmlBean.java ( 1 )

```
32    String url = "jdbc:postgresql://" + ip + ":5432/" + db;
33    Connection connection =
34        DriverManager.getConnection(url, "postgres", "postgres");
35    // [3] SQL文のコンパイル作成
36    Statement stmt = connection.createStatement();
37    // [4] SQL文を実行する
38    // executeUpdateは処理結果の行数が返ってきます
39    int plus = stmt.executeUpdate(
40        "update t_お気に入り set お気に入り数 = お気に入り数+1 where 店舗番号 = " + inputNumber);
41    ResultSet rs = stmt.executeQuery(
42        "select * from v_お気に入り店舗 "
43        + "where 店舗番号 = " + inputNumber + " order by 店舗番号");
44    // [5] RDBの検索結果を取り出す
45    resultSet = resultSet + "<table border=1>";
46    resultSet = resultSet + "<tr><th>店舗番号</th><th>店舗名</th><th>市町村区名</th>";
47    resultSet = resultSet + "<th>住所</th><th>電話番号</th><th>特売情報</th><th>お気に入り数</th></tr>";
48
49    while (rs.next()) {
50        number = rs.getInt("店舗番号");
51        name = rs.getString("店舗名");
52        city = rs.getString("市町村区名");
53        address = rs.getString("住所");
54        tel = rs.getString("電話番号");
55        tokubai = rs.getString("特売情報");
56        okiniiri = rs.getInt("お気に入り数");
57        resultSet = resultSet + "<tr><td>" + number + "</td><td>" + name + "</td><td>" + city + "</td><td>";
58        resultSet = resultSet + address + "</td><td>" + tel + "</td><td>" + tokubai + "</td><td>" + okiniiri + "</td></tr>";
59    }
60    resultSet = resultSet + "</table>";
```

図15. SunkusUpdateHtmlBean.java ( 2 )

```
61    // [6] データベースから切断する
62    stmt.close();
63    connection.close();
64
65    catch (Exception e) {
66        e.printStackTrace();
67    }
68
69
70
71    public void setInputNumber(int inputNumber) {
72        this.inputNumber = inputNumber;
73    }
74
75    public int getInputNumber() {
76        return inputNumber;
77    }
78    public String getResultSet() {
79        return resultSet;
80    }
81
82
83
```

図16. SunkusUpdateHtmlBean.java ( 3 )

## 2. リクエストコントロール用Servletクラスを作成します

Eclipseパッケージ・エクスプローラより

sunkusfeプロジェクトを右クリック→新規→その他→Web→サーブレット

→以下の内容で実装する

- SunkusServlet.java

パッケージ : jp.ict.aso.controller

クラス名 : SunkusServlet

ソースコード例 : 以下のように実装する ※アノテーションは/Sunkus

【処理概要】

doPost()メソッド内ではリクエストパラメータactionの値により処理を振り分けます。

- actionの値がviewの場合は、SunkusHtmlBeanを実体化して指定対象の設定で検索を行い結果をリクエストスコープに保存してsunkusView.jspへフォワードします。
- actionの値がupdateの場合は、SunkusUpdateHtmlBeanを実体化して指定対象の設定で更新を行い結果をリクエストスコープに保存してsunkusUpdate.jspへフォワードします。

```
1 package jp.ict.aso.controller;
2
3 import java.io.IOException;
4
5 import javax.servlet.RequestDispatcher;
6 import javax.servlet.ServletException;
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
12 import jp.ict.aso.model.SunkusHtmlBean;
13 import jp.ict.aso.model.SunkusUpdateHtmlBean;
14
15 @WebServlet("/Sunkus")
16 public class SunkusServlet extends HttpServlet {
17
18     protected void doGet(HttpServletRequest request,
19         HttpServletResponse response)
20         throws ServletException, IOException {
21
22         //最初は全件出力
23         SunkusHtmlBean shb = new SunkusHtmlBean("192.168.56.200", "mrp"); //実体化
24         shb.setInputCity("福岡県"); //全件出力用
25         shb.sunkusSearch(); //検索メソッド実行
26         // リクエストスコープに保存
27         request.setAttribute("result", shb);
28         //////////////////////////////////////
29         // フォワード
```

図17. SunkusServlet.java (1)

```
30     RequestDispatcher dispatcher =
31         request.getRequestDispatcher(
32             "/WEB-INF/jsp/sunkusView.jsp");
33     dispatcher.forward(request, response);
34 }
35 protected void doPost(HttpServletRequest request,
36     HttpServletResponse response)
37     throws ServletException, IOException {
38
39     request.setCharacterEncoding("UTF-8");
40
41     // サーブレットクラスの動作を決定する「action」の値を
42     // リクエストパラメータから取得
43     String action = request.getParameter("action");
44     // 参照 (入力値をプロパティに設定)
45     // リクエストパラメータの取得
46     String city = request.getParameter("city"); //koko
47
48     if(action.equals("view")){ //actionが表示のとき
49         SunkusHtmlBean shb = new SunkusHtmlBean("192.168.56.200", "mrp"); //実体化
50         shb.setInputCity(city); //市町村区名をセット
51         shb.sunkusSearch(); //検索メソッド実行
52         // リクエストスコープに保存
53         request.setAttribute("result", shb);
54         //////////////////////////////////////
55         // フォワード
56         RequestDispatcher dispatcher =
57             request.getRequestDispatcher(
58                 "/WEB-INF/jsp/sunkusView.jsp");
59         dispatcher.forward(request, response);
60     }
```

図18. SunkusServlet.java (2)

```

61     else if(action.equals("update")){ //actionが更新のとき
62         // 更新（入力値をプロパティに設定）
63         // リクエストパラメータの取得と数値変換
64         String number = request.getParameter("number");
65         int number = Integer.parseInt(number); //numberの数値変換
66
67         SunkusUpdateHtmlBean sub
68             = new SunkusUpdateHtmlBean("192.168.56.200","mrp");//実体化
69         sub.setInputNumber(number); //店番号セット
70         sub.sunkusUpdate(); //更新メソッド実行
71         // リクエストスコープに保存
72         request.setAttribute("result", sub);
73         ////////////////////////////////////////
74         // フォワード
75         RequestDispatcher dispatcher =
76             request.getRequestDispatcher
77                 ("/WEB-INF/jsp/sunkusUpdate.jsp");
78         dispatcher.forward(request, response);
79     }
80 }
81 }

```

図19. SunkusServlet.java（3）

## 2. お気に入り店舗検索用画面のJSPファイルを作成します

Eclipseパッケージ・エクスプローラより

sunkusfeプロジェクトを右クリック→新規→その他→Web→JSPファイル

→以下の内容で作成する

・ sunkusView.jsp

保存場所：sunkusfe/src/main/webapp/**WEB-INF/jsp** ← 注意！

ファイル名：sunkusView.jsp

ソースコード例：以下の内容で実装する

```

1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8" import="jp.ict.aso.model.*, java.util.*" %>
3  <%
4      SunkusHtmlBean shb=(SunkusHtmlBean) request.getAttribute("result");
5  %>
6  <html>
7  <head>
8  <title>コンビニ</title>
9  <link rel="stylesheet" type="text/css" href="sunkus.css">
10 </head>
11 <body background="images/abstract-258339_1280.png">
12 <center>
13 <h1>コンビニ店舗検索システム</h1>
14 <h2>市区町村名をもとにコンビニ店舗情報を検索します</h2>
15 <form method="POST" action="Sunkus?action=view">
16 <p>市区町村名：<input type="text" name="city" value="春日市"></p>
17 <input type="submit" value="検索">
18 </form>
19 <hr>
20 <h2>店舗番号をもとにお気に入り情報を更新します</h2>
21 <form method="POST" action="Sunkus?action=update">
22 <p>店舗番号：<input type="number" name="number" value="608239"></p>
23 <input type="submit" value="更新">
24 </form>
25 <hr>
26 <p>
27 検索結果：<%=shb.getResultSet() %><br>
28 </p>
29
30 <a href="Sunkus">はじめる</a>
31
32 <hr>
33 </center>
34 </body>
35 </html>

```

図20. sunkusView.jsp

## 3. お気に入り数更新用画面のJSPファイルを作成します

Eclipseパッケージ・エクスプローラより

sunkusfeプロジェクトを右クリック→新規→その他→Web→JSPファイル

→以下の内容で作成する

- sunkusUpdate.jsp

保存場所：sunkusfe/src/main/webapp/**WEB-INF/jsp** ← **注意！**

ファイル名：sunkusUpdate.jsp

ソースコード例：以下の内容で実装する

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
2   pageEncoding="UTF-8" import="jp.ict.aso.model.*, java.util.*" %>
3 <%
4   SunkusUpdateHtmlBean sub=(SunkusUpdateHtmlBean) request.getAttribute("result");
5 %>
6 <html>
7 <head>
8   <title>コンビニ</title>
9   <link rel="stylesheet" type="text/css" href="sunkus.css">
10 </head>
11 <body background="images/abstract-258339_1280.png">
12 <center>
13   <h1>コンビニ店舗検索システム</h1>
14   <h2>市区町村名をもとにコンビニ店舗情報を検索します</h2>
15   <form method="POST" action="Sunkus?action=view">
16     <p>市区町村名:<input type="text" name="city" value="春日市"></p>
17     <input type="submit" value="検索">
18   </form>
19 <hr>
20   <h2>店舗番号をもとにお気に入り情報を更新します</h2>
21   <form method="POST" action="Sunkus?action=update">
22     <p>店舗番号:<input type="number" name="number" value="608239"></p>
23     <input type="submit" value="更新">
24   </form>
25 <hr>
26 <p>
27   更新結果:<%=sub.getResultSet() %><br>
28 </p>
29
30 <a href="Sunkus">はじめる</a>
31
32 <hr>
33 </center>
34 </body>
35 </html>
```

図21. sunkusUpdate.jsp

## 実行確認

サーブレットクラス（SunkusServlet.java）を実行します。

### 1. 正常に実行されることを確認します

以下のようにデフォルト設定の「春日市」で検索されて、お気に入り店舗を更新できることを確認します。

| 店舗番号   | 店舗名       | 市町村区名  | 住所         | 電話番号         | 特売情報  | お気に入り数 |
|--------|-----------|--------|------------|--------------|-------|--------|
| 608239 | 春日桜ヶ丘八丁目店 | 福岡県春日市 | 桜ヶ丘八丁目1-1  | 092-592-5578 | アイスケー | 2      |
| 608454 | 春日小倉二丁目店  | 福岡県春日市 | 小倉二丁目134-1 | 092-588-6388 | ブランドー | 1      |
| 608584 | 春日白水ヶ丘店   | 福岡県春日市 | 白水ヶ丘四丁目26番 | 092-589-3056 | ビール   | 0      |

図22. お気に入り店舗検索更新画面

### 2. スタイルシートや背景画像で簡単に装飾します

ソースコード例として提示したJSPにはCSSファイルが指定してあるので簡単に実装します。

---

Eclipseパッケージ・エクスプローラより  
sunkusfeプロジェクトを右クリック→新規→CSSファイル  
→以下の内容で作成する

---

- sunkus.css

保存場所：sunkusfe/src/main/webapp/**WEB-INF/** ← **注意！**

ファイル名：sunkus.css

ソースコード例：以下の内容で実装する

```
1 @charset "UTF-8";
2 body {
3     background-color: #bbb;
4     font-size: 14px;
5     color: #555;
6     a:link, a:visited {
7         color: #844;
8     }
9     a:hover {
10        color: #faa;
11    }
12    a {
13        text-decoration: none;
14    }
15    p, form {
16        margin: 0;
17        padding: 4px 0;
18    }
19    div {
20        font-weight: bold;
21    }
22    h1 {
23        margin: 10px 0;
24        padding: 0;
25        font-size: 38px;
26        text-align: center;
27    }
28    h2 {
29        margin: 10px 0;
30        padding: 0;
31        font-size: 16px;
32        text-align: center;
33    }
34    table {
35        table-layout: fixed;
36        width: 740px;
37        font-size: 12px;
38        margin: 15px auto;
39        padding: 0;
40        background-color: #fff;
41        border-collapse: collapse;
42    }
43    th {
44        text-align: center;
45        background-color: #ffd78c;
46    }
47    th, td {
48        margin: 0;
49        padding: 6px 16px;
50        border-bottom: 12px solid #bbb;
51    }
```

図23. sunkus.css

あわせて背景画像を設定します。

---

【参考】フリー画像サイト↓

<https://pixabay.com/illustrations/search/background/>

---

- images/背景画像

画像を配置するimagesフォルダの位置は以下の場所です。

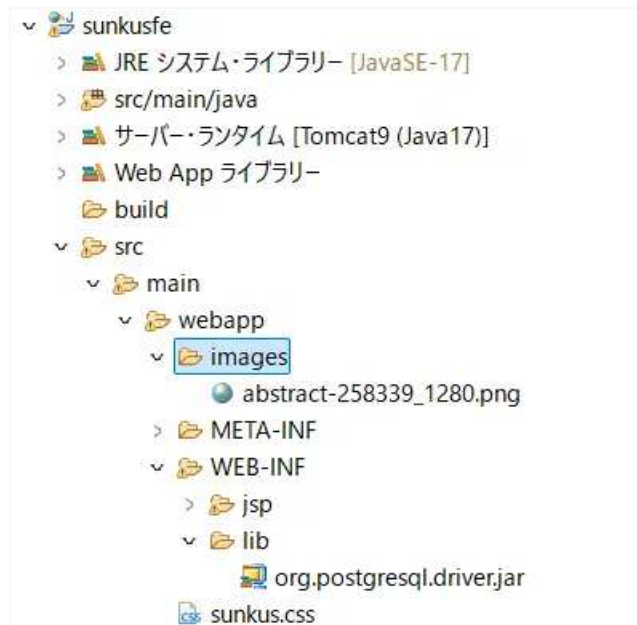


図24. 背景画像の配置場所

imagesフォルダを配置後、jspファイルのbodyタグにあるbackground属性を変更して背景画像を有効にしてください。

### 3. 装飾を確認します

再度サーブレットクラス（SunkusServlet.java）を実行し画面の装飾を確認します。



図25. 装飾後のお気に入り店舗検索更新画面

## 仕様反映

設計のもとになった基本情報技術者試験の問題では『指定した市区町村にある店舗情報をスマートフォンを用いて検索するシステムを構築する』となっていました。



これを受けて外部設計では『スマートフォンでの運用が前提のため絞り込みや選択にはドロップダウンリストを用いレコード毎に更新ボタンを設置します』としていましたが、当初は内部設計において実装を単純化するためにこの仕様は『仕様反映で実現します』としていました。

そこで当初の内部設計どおりに以下のMVCモデル図の枠の背景が赤と青のクラスを追加設計して実装します。

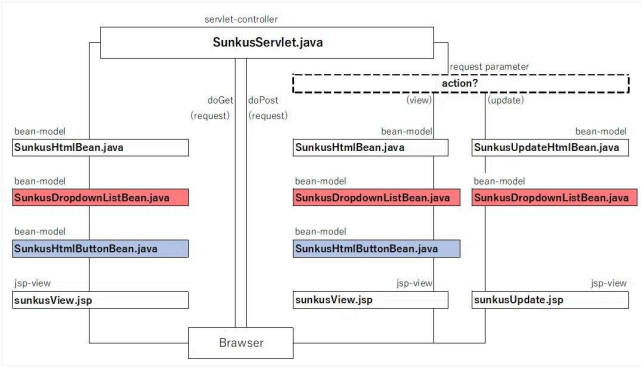


図26. MVCモデル図（クラス連携図）

# 1. SunkusDropdownListBean.classの設計を追加します

(仕様)

- データベースから店舗番号の**検索と表示**を行います。
- オーバーロードされたコンストラクタを3つ用意しSQL文を指定します。
  - ①IPとDB名のみを指定した場合はすべての店舗番号を検索するSQL文
  - ②IPとDB名と市区町村名を指定した場合は対応する市区町村の店舗番号を検索するSQL文。
  - ③IPとDB名と店舗番号を指定した場合は対応する店舗番号を検索するSQL文。
- tenpoSearch()メソッドを実行することで検索が実行され結果がhtml形式で表組整形されてresultSetフィールドに格納されます。
- 利用側のクラスはgetResultSet()メソッドで店舗番号を取得します。

(クラス図)

|                                                                                                                                                                                                                                          |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| package: jp.ict.aso.model<br>SunkusDropdownListBean                                                                                                                                                                                      |
| - ip: String<br>- db: String<br>- number: String<br>- city: String<br>- sql: String<br>- resultSet: String                                                                                                                               |
| + SunkusDropdownListBean(ip: String, db: String)<br>+ SunkusDropdownListBean(city: String, ip: String, db: String)<br>+ SunkusDropdownListBean(number: int, ip: String, db: String)<br>+ tenpoSearch(): void<br>+ getResultSet(): String |
| -:private   +:public   #:protected                                                                                                                                                                                                       |



## 2. SunkusHtmlButtonBean.classの設計を追加します

(仕様)

- ・データベースから店舗情報を**検索し更新ボタン付きで表示**を行います。  
(SunkusHtmlBeanクラスに更新ボタンを付加したクラスです)
- ・コンストラクタの引数でPostgreSQLサーバのIPとDB名を指定して実体化します。
- ・setInputCity()メソッドに市区町村名をセットしてsunkusSearch()メソッドを実行することで対象となる地域の店舗情報がDBから検索されます。
- ・検索結果はhtml形式で表組整形されてresultSetフィールドに格納されます。**この時、更新ボタンも付加します。**
- ・利用側のクラスはgetResultSet()メソッドで検索結果を取得します。

(クラス図)

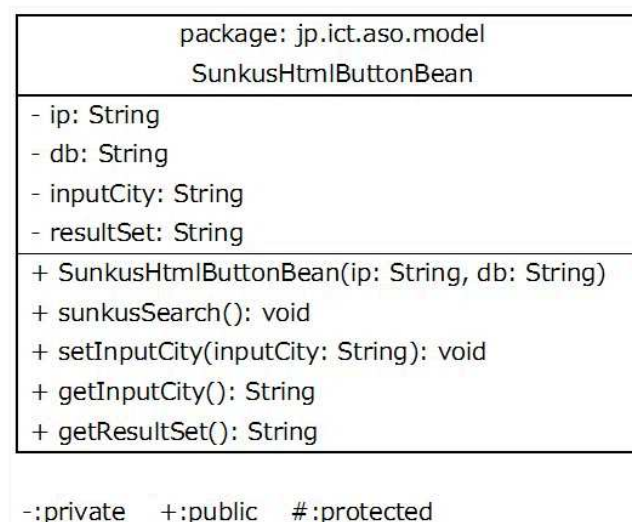


図28. SunkusHtmlButtonBeanクラス図

## 3. 追加設計したJavaBeansクラスを実装します

Eclipseパッケージ・エクスプローラより  
sunkusfeプロジェクトを右クリック→新規→クラス  
→以下の内容で作成する

- ・ SunkusDropDownListBean.java  
   パッケージ : jp.ict.aso.model  
   クラス名 : SunkusDropDownListBean  
   ソースコード例 : 図27のクラス図より以下のように実装する

```

1 package jp.ict.aso.model;
2 import java.io.Serializable;
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.Statement;
7
8 public class SunkusDropDownListBean implements Serializable{
9     private String ip;
10    private String db;
11    private String number;
12    private String city;
13    private String sql;
14    private String resultSet;
15
16    public SunkusDropDownListBean(String ip, String db){
17        this.ip=ip;
18        this.db=db;
19        resultSet="";
20        sql="select 店舗番号 from t_店舗 order by 店舗番号";
21    }
22    public SunkusDropDownListBean(String city, String ip, String db){
23        this.ip=ip;
24        this.db=db;
25        resultSet="";
26        this.city=city;
27        sql="select 店舗番号,市区町村名 from v_お気に入り店舗 "
28            + "where 市区町村名 like '%" + city + "%' order by 店舗番号";
29    }
30    public SunkusDropDownListBean(int number, String ip, String db){
31        this.ip=ip;
32        this.db=db;
33        resultSet="";
34        this.number=String.valueOf(number);
35        sql="select 店舗番号 from v_お気に入り店舗 "
36            + "where 店舗番号 = '"+ number + "' order by 店舗番号";
37    }

```

図29. SunkusDropDownListBean.java（1）

```

38
39 public void tenpoSearch(){
40     try{
41         /*
42          * ●業務処理セクション
43          */
44         // [1] JDBCドライバのロード
45         Class.forName("org.postgresql.Driver");
46         // [2] RDBへの接続
47         String url="jdbc:postgresql://" + ip + ":5432/" + db;
48         Connection connection =
49             DriverManager.getConnection(url,"postgres","postgres");
50         // [3] SQL文のコンテナ作成
51         Statement stmt = connection.createStatement();
52         // [4] SQL文を実行する
53         ResultSet rs = stmt.executeQuery(sql);
54         // [5] RDBの検索結果を取り出す
55         resultSet=resultSet+"<select name='"+number+"'>";
56         while(rs.next()){
57             number = rs.getString("店舗番号");
58             resultSet=resultSet+"<option>" + number + "</option>";
59         }
60         resultSet=resultSet+"</select>";
61         // [6] データベースから切断する
62         stmt.close();
63         connection.close();
64     }
65     catch (Exception e) {
66         e.printStackTrace();
67     }
68 }
69 public String getResultSet() {
70     return resultSet;
71 }
72 }

```

図30. SunkusDropDownListBean.java（2）

---

Eclipseパッケージ・エクスプローラより  
sunkusfeプロジェクトを右クリック→新規→クラス  
→以下の内容で作成する

---

- SunkusHtmlButtonBean.java  
パッケージ : jp.ict.aso.model  
クラス名 : SunkusHtmlButtonBean  
ソースコード例 : 図28のクラス図より以下のように実装する

```

1 package jp.ict.aso.model;
2
3 import java.io.Serializable;
4 import java.sql.Connection;
5 import java.sql.DriverManager;
6 import java.sql.ResultSet;
7 import java.sql.Statement;
8
9 public class SunkusHtmlButtonBean implements Serializable{
10     private String ip;
11     private String db;
12     private String inputCity;
13
14     private String resultSet;
15
16     public SunkusHtmlButtonBean(String ip, String db){
17         resultSet="";
18         inputCity="";
19         this.ip=ip;
20         this.db=db;
21     }
22     public void sunkusSearch() {
23         int number;
24         String name;
25         String city;
26         String address;
27         String tel;
28         String tokubai;
29         int okiniiri;
30     try{

```

図31. SunkusHtmlButtonBean.java ( 1 )

```

31 // [1]JDBCドライバのロード
32 Class.forName("org.postgresql.Driver");
33 // [2]RDBへの接続
34 String url="jdbc:postgresql://"+ip+":5432/"+db;
35 Connection connection =
36     DriverManager.getConnection(url, "postgres", "postgres");
37 // [3]SQL文のコンテナ作成
38 Statement stmt = connection.createStatement();
39 // System.out.println("select * from t_店舗 where 市区町村名 like '%" + this.inputCity + "%");
40 // [4]SQL文を実行する
41 ResultSet rs = stmt.executeQuery
42     ("select * from v_お気に入り店舗 "
43      + "where 市区町村名 like '%" + inputCity + "%' order by 店舗番号");
44 // [5]RDBの検索結果を取り出す
45 resultSet=resultSet+"<table border=1>";
46 resultSet=resultSet+"<tr><th>店舗番号</th><th>店舗名</th><th>市区町村名</th>";
47 resultSet=resultSet+"<th>住所</th><th>電話番号</th><th>特売情報</th>"; //koko
48 resultSet=resultSet+"<th colspan=2>お気に入り数</th></tr>"; //koko
49
50 while ( rs.next() ){
51     number = rs.getInt("店舗番号");
52     name = rs.getString("店舗名");
53     city = rs.getString("市区町村名");
54     address = rs.getString("住所");
55     tel = rs.getString("電話番号");
56     tokubai = rs.getString("特売情報");
57     okiniiri = rs.getInt("お気に入り数");
58 }

```

図32. SunkusHtmlButtonBean.java ( 2 )

```

59     resultSet=resultSet+"<form method="+POST+" action="+Sunkus?action=update+">"; //koko
60     resultSet=resultSet+"<tr><td>"+number + "</td><td> "+name+ "</td><td>"+city+"</td><td>";
61     resultSet=resultSet+address+"</td><td>"+tel+"</td><td>"+tokubai+"</td><td>"+okiniiri+"</td>";
62     resultSet=resultSet+"<td><input type="+hidden+" name="+number+" value="+number+">"; //koko
63     resultSet=resultSet+"<input type="+submit+" value="+更新+"></td></tr></form>";
64 }
65
66 resultSet=resultSet+"</table>";
67
68 // [6]データベースから切断する
69 stmt.close();
70 connection.close();
71 catch ( Exception e) {
72     e.printStackTrace();
73 }
74 }
75
76 public void setInputCity(String inputCity) {
77     this.inputCity = inputCity;
78 }
79 public String getInputCity() {
80     return inputCity;
81 }
82 public String getResultSet() {
83     return resultSet;
84 }
85
86 }

```

図33. SunkusHtmlButtonBean.java ( 3 )

## 4. リクエストコントロール用Servletクラスを修正します

### SunkusServlet.java

元のソースコードの2 8行目に①のブロックを， 5 4行目に②のブロックを， 7 3行目に③のブロックを追加します。

```

1 package jp.ict.aso.controller;
2
3 import java.io.IOException;
4
5 import javax.servlet.RequestDispatcher;
6 import javax.servlet.ServletException;
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
12 import jp.ict.aso.model.SunkusDropDownListBean;
13 import jp.ict.aso.model.SunkusHtmlBean;
14 import jp.ict.aso.model.SunkusHtmlButtonBean;
15 import jp.ict.aso.model.SunkusUpdateHtmlBean;
16
17 @WebServlet("/Sunkus")
18 public class SunkusServlet extends HttpServlet {
19
20     protected void doGet(HttpServletRequest request,
21         HttpServletResponse response)
22         throws ServletException, IOException {
23
24         //最初は全件出力
25         SunkusHtmlBean shb = new SunkusHtmlBean("192.168.56.200", "mrp"); //実体化
26         shb.setInputCity("福岡県"); //全件出力用
27         shb.sunkusSearch(); //検索メソッド実行
28         // リクエストスコープに保存
29         request.setAttribute("result", shb);

```

図34. SunkusServlet.java ( 1 )

```

30 //①//////////////////////////////////////
31 //店舗番号ドロップダウンリスト出力
32 SunkusDropDownListBean sdl =
33     new SunkusDropDownListBean("192.168.56.200", "mrp"); //実体化
34 sdl.tenpoSearch(); //店舗出カメソッド実行
35 //リクエストスコープに保存
36 request.setAttribute("tenpo", sdl);
37 //////////////////////////////////////////////////////////////
38 //ボタン付き
39 SunkusHtmlButtonBean shbb =
40     new SunkusHtmlButtonBean("192.168.56.200", "mrp"); //実体化
41 shbb.setInputCity("福岡県"); //市町村区名をセット
42 shbb.sunkusSearch(); //検索メソッド実行
43 //リクエストスコープに保存
44 request.setAttribute("resultButton", shbb);
45 //////////////////////////////////////////////////////////////
46 // フォワード
47 RequestDispatcher dispatcher =
48     request.getRequestDispatcher
49         ("/WEB-INF/jsp/sunkusView.jsp");
50 dispatcher.forward(request, response);
51 }
52
53 protected void doPost(HttpServletRequest request,
54     HttpServletResponse response)
55     throws ServletException, IOException {
56
57     request.setCharacterEncoding("UTF-8");
58

```

図35. SunkusServlet.java ( 2 )

```

59 // サブレットクラスの動作を決定する「action」の値を
60 // リクエストパラメータから取得
61 String action = request.getParameter("action");
62 // 参照（入力値をプロパティに設定）
63 // リクエストパラメータの取得
64 String city = request.getParameter("city"); //koko
65
66 if(action.equals("view")){ //actionが表示のとき
67     SunkusHtmlBean shb = new SunkusHtmlBean("192.168.56.200", "mrp"); //実体化
68     shb.setInputCity(city); //市町村区名をセット
69     shb.sunkusSearch(); //検索メソッド実行
70     // リクエストスコープに保存
71     request.setAttribute("result", shb);
72 //②//////////////////////////////////////
73 //店舗番号ドロップダウンリスト出力
74 SunkusDropDownListBean sdl =
75     new SunkusDropDownListBean(city, "192.168.56.200", "mrp"); //実体化
76 sdl.tenpoSearch(); //店舗出カメソッド実行
77 //リクエストスコープに保存
78 request.setAttribute("tenpo", sdl);
79 //////////////////////////////////////////////////////////////
80 //ボタン付き
81 SunkusHtmlButtonBean shbb =
82     new SunkusHtmlButtonBean("192.168.56.200", "mrp"); //実体化
83 shbb.setInputCity(city); //市町村区名をセット
84 shbb.sunkusSearch(); //検索メソッド実行
85 //リクエストスコープに保存
86 request.setAttribute("resultButton", shbb);
87 //////////////////////////////////////////////////////////////
88 // フォワード
89 RequestDispatcher dispatcher =
90     request.getRequestDispatcher
91         ("/WEB-INF/jsp/sunkusView.jsp");
92 dispatcher.forward(request, response);
93

```

図36. SunkusServlet.java ( 3 )

```

94 }
95 else if(action.equals("update")){ //actionが更新のとき
96     // 更新（入力値をプロパティに設定）
97     // リクエストパラメータの取得と数値変換
98     String number = request.getParameter("number");
99     int number = Integer.parseInt(snumber); //numberの数値変換
100
101     SunkusUpdateHtmlBean sub = new SunkusUpdateHtmlBean("192.168.56.200", "mrp"); //実体化
102     sub.setInputNumber(number); //店番号セット
103     sub.sunkusUpdate(); //更新メソッド実行
104     // リクエストスコープに保存
105     request.setAttribute("result", sub);
106 //③//////////////////////////////////////
107 //店舗番号ドロップダウンリスト出力
108 SunkusDropDownListBean sdl =
109     new SunkusDropDownListBean(number, "192.168.56.200", "mrp"); //実体化
110 sdl.tenpoSearch(); //店舗出カメソッド実行
111 //リクエストスコープに保存
112 request.setAttribute("tenpo", sdl);
113 //////////////////////////////////////////////////////////////
114 // フォワード
115 RequestDispatcher dispatcher =
116     request.getRequestDispatcher
117         ("/WEB-INF/jsp/sunkusUpdate.jsp");
118 dispatcher.forward(request, response);
119 }
120 }
121 }

```

図37. SunkusServlet.java ( 4 )

## 5. お気に入り店舗検索用画面のJSPファイルを修正します

### sunkusView.jsp

市町村区名はhtmlタグでjsp内に直接ハードコーディングします。  
店舗番号は市町村区選択により動的に変化するためDBから読み込みます。

kokoの部分を追加修正してください。4か所あります。

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
2 <%@ pageEncoding="UTF-8" import="jp.ict.aso.model.*, java.util.*" %>
3 <%
4 //koko
5 SunkusHtmlButtonBean shbb=(SunkusHtmlButtonBean)request.getAttribute("resultButton");
6 SunkusDropDownListBean sdl=(SunkusDropDownListBean)request.getAttribute("tenpo");
7 %>
8 <html>
9 <head>
10 <title>コンビニ</title>
11 <link rel="stylesheet" type="text/css" href="sunkus.css">
12 </head>
13 <body background="images/abstract-258339_1280.png">
14 <center>
15 <h1>コンビニ店舗検索システム</h1>
16 <h2>市区町村名をもとにコンビニ店舗情報を検索します</h2>
17 <form method="POST" action="Sunkus?action=view">
18 <!--koko--><p>市区町村名:<input type="text" name="city" value="春日市"></p> -->
19 <p>市区町村名:<select name="city">
20 <option>福岡市中央区</option>
21 <option>福岡市博多区</option>
22 <option>福岡市城南区</option>
23 <option>福岡市早良区</option>
24 <option>福岡市南区</option>
25 <option>福岡市東区</option>
26 <option>大野城市</option>
27 <option>筑紫野市</option>
28 <option>久留米市</option>
29 <option>北九州市</option>
30 <option>春日市</option>
31 <option>刈田町</option>
32 <option value="">全地域</option>
33 </select>
34 <!-- -->
35 <input type="submit" value="検索">
36 </form>
37 </center>
```

図38. sunkusView.jsp (1)

```
38 <h2>店舗番号をもとにお気に入り情報を更新します</h2>
39 <form method="POST" action="Sunkus?action=update">
40 <!--koko--><p>店舗番号:<input type="number" name="number" value="608239"></p> -->
41 <p>店舗番号:<%=sdl.getResultSet(0)%>
42 <input type="submit" value="更新">
43 </form>
44 </center>
45 <!--koko 変更-->
46 <p>
47 <b><%=shbb.getInputCity(0)%></b>
48 検索結果:<%=shbb.getResultSet(0)%><br>
49 </p>
50 <!-- -->
51 <a href="Sunkus">はじめに戻る</a>
52
53 <hr>
54 </center>
55 </body>
56 </html>
```

図39. sunkusView.jsp (2)

## 6. お気に入り数更新用画面のJSPファイルを修正します

### sunkusUpdate.jsp

市町村区名はhtmlタグでjsp内に直接ハードコーディングします。  
店舗番号は市町村区選択により動的に変化するためDBから読み込みます。

kokoの部分を追加修正してください。3か所あります。

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8" import="jp.ict.aso.model.*, java.util.*" %>
3
4 //koko
5 SunkusUpdateHtmlBean sub=(SunkusUpdateHtmlBean)request.getAttribute("result");
6 SunkusDropDownListBean sd1=(SunkusDropDownListBean)request.getAttribute("tenpo");
7
8 <html>
9 <head>
10 <title>コンビニ</title>
11 <link rel="stylesheet" type="text/css" href="sunkus.css">
12 </head>
13 <body background="images/abstract-258339_1280.png">
14 <center>
15 <h1>コンビニ店舗検索システム</h1>
16 <h2>市区町村名をもとにコンビニ店舗情報を検索します</h2>
17 <form method="post" action="Sunkus?action=view">
18 <!--koko-->市区町村名:<input type="text" name="city" value="春日市"></p> -->
19 <p>市区町村名:<select name="city">
20 <option>福岡市中央区</option>
21 <option>福岡市博多区</option>
22 <option>福岡市城南区</option>
23 <option>福岡市早良区</option>
24 <option>福岡市南区</option>
25 <option>福岡市東区</option>
26 <option>大野城市</option>
27 <option>筑紫野市</option>
28 <option>久留米市</option>
29 <option>北九州市</option>
30 <option>春日市</option>
31 <option>河内町</option>
32 <option value="">全地域</option>
33 </select>
34 <!-- -->
35 <input type="submit" value="検索">
36 </form>
37 </hr>

```

図40. sunkusUpdate.jsp (1)

```

38 <h2>店舗番号をもとにお気に入り情報を更新します</h2>
39 <!--><form method="POST" action="Sunkus?action=update">
40 <!--koko--><p>店舗番号:<input type="number" name="number" value="608239"></p> -->
41 <p>店舗番号:<%=sd1.getResultSet(0)%>
42 <input type="submit" value="更新">
43 </form>
44 <hr>
45 <p>
46 更新結果:<%=sub.getResultSet(0)%><br>
47 </p>
48
49 <a href="Sunkus">はじめる</a>
50
51 <hr>
52 </center>
53 </body>
54 </html>

```

図41. sunkusUpdate.jsp (2)

## 最終確認

ドロップダウンリスト及び更新ボタンが有効に働くか確認して完成です。



図42. コンビニ店舗検索システム完成