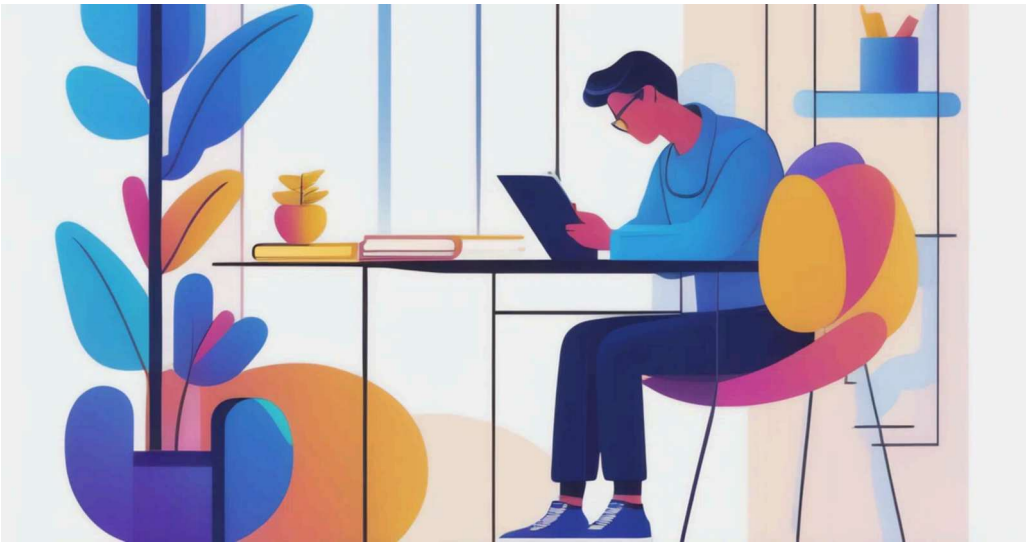

MVCモデルによるWebアプリケーション開発（No８.ポーカーゲーム）-EE8

office・M
2024年1月13日 18:02

¥1,000 •••

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。今回は部品として作成されたクラスファイル（ポーカーゲームのカード処理行うBean）を活用する方法を学びます。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

CardConfirmHtml5Bean.classの使い方とクラス図

CardScoreBean.classの使い方とクラス図

実装準備

1．ポーカーロジック用JavaBeansクラスを使用する準備をします

2．カードの画像用のフォルダを作成します

実装手順

1．リクエストコントロール用Servletクラスを作成します

2．初期配布状態の画面のJSPファイルを作成します

3．役確定画面のJSPファイルを作成します

実行確認

1．InternalServerErrorとなります

2．実行用Beanを配置します

3．再度実行します

テスト

仕様反映

ソースコード例と提供Bean

1．CardShuffleArrayListBean.class

2．CardShuffleHtml10Bean.class

3．CardHanteiHtml5Bean.class

4．CardConfirmHtml5Bean.class

5．CardScoreBean.class

6．カードイメージ

7．PokerServlet.java

8．poker10.jsp

9．poker05.jsp

10．PokerScoreTest.java

連絡先

開発概要

JavaEE8アプリケーション・サーバ（Tomcat）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。

題材としてポーカーゲームを行うWebアプリケーションを作成します。52枚のカードをシャッフルして10枚取り出し、5枚のカードで役を判定します。役の状況を判断してカード構成は１回だけ交換が可能です。カードのシャッフル、取り出し、役の判定等のカード処理を行うBeanは提供します。

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）ーView（表示）ー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、業務ロジックであるBeanは作成しません。他のチームメンバーが作成したという想定で提供されたBeanを活用することにします。

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 Eclipse 2024

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。

開発に必要な仕様と最小限の設計ドキュメントは提示します。この情報をもとに、まずは、自分で試行錯誤しながら実装してみてください。

設計仕様

【ポーカーゲームの仕様】
以下の条件を実現するWebアプリケーションの作成を行います。
（条件）
・urlを指定してサイトにアクセスすると52枚のカードをシャッフルし10枚取り出します。５枚は表向きで、残りの5枚は裏向きで表示します。これを初期配布とします。
・初期配布の状態で表向き５枚の役を判定し表示します。また持ち点と掛け点のテキストボックス、カード交換のチェックボックス及びカード交換のボタンを表示します。
・カード構成は１回だけ交換可能です。役の状態を判断して表向きの残したいカードと裏向きの残したいカードをチェックボックスで指定します。
・「カード交換ボタン」をクリックすると残したカードの役が確定され払い戻しの点数が計算されます。
・役の倍率や処理の流れのイメージは図1.1、図1.2を参照してください。

図1.1 初期配布イメージ図

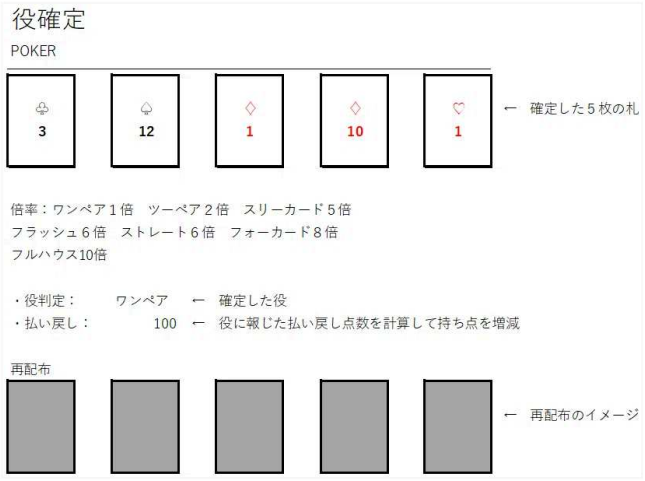

図1.2 役確定イメージ図

【Model（処理ロジック）】
CardShuffleArrayListBean.classが提供されます。
（仕様）
・52枚のカードをシャッフルします。結果はArrayListクラスに保存します。

CardShuffleHtml10Bean.classが提供されます。
（仕様）
・シャッフル済みの52枚のカードの最初から10枚を取り出し5枚は表向きで残りの5枚は裏向きに設定します。
・カード交換時に残すカードを指定できるよう チェックボックスを付加します。

CardHanteiHtml5Bean.classが提供されます。
（仕様）
・初期配布状態の表向きの5枚のカードの役をCardScoreBeanを用いて判定し設定します。
・リクエストスコープに10枚のカードと５枚の役の判定結果のインスタンスを保存しjspにフォワードします

CardConfirmHtml5Bean.classが提供されます。
（仕様）
・チェックされた5枚のカードの役をCardScoreBeanを用いて判定し設定します。
・リクエストスコープに5枚のカードの確定した役と払い戻し点数のインスタンスを保存しjspにフォワードします

CardScoreBean.classが提供されます。
（仕様）
・5枚のカードの役を判定し払い戻し点数を計算します。役と払い戻しの倍率は以下のようになります。
・ワンペア１倍、ツーペア２倍、スリーカード５倍、フラッシュ６倍、ストレート６倍、フォーカード８倍、フルハウス10倍

外部設計

接続urlは**http://localhost:8080/game/Poker**とします。よってEclipseの動的Webプロジェクトの名前は**game**となり、サーブレットのurlパターンは**Poker**となります。

Eclipseのメニューバーより
ファイル→新規→動的Webプロジェクト→「game」プロジェクトを作成する
→図2の内容で設定する

図2.　動的Webプロジェクトの設定

※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３のような、起動画面が表示されます。初期配布の状態の役を見て残すカードを裏向きも含めてチェックします。「カード交換ボタン」をクリックすると、図４のように確定した５枚のカードの役が判定され、払い戻しの点数が計算されます。

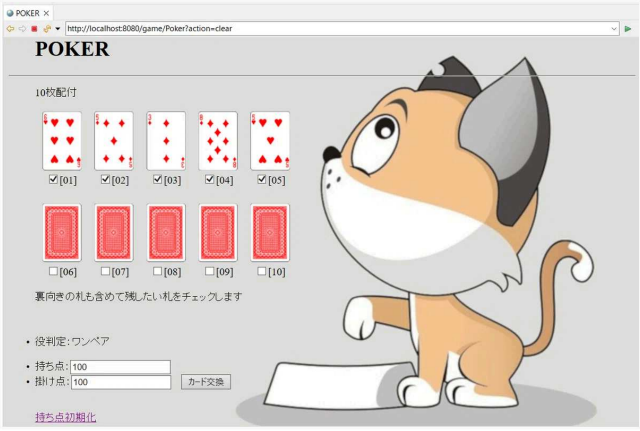

図３．起動画面

図４．カード確定画面

内部設計

クラス連携図は図５のようになります。

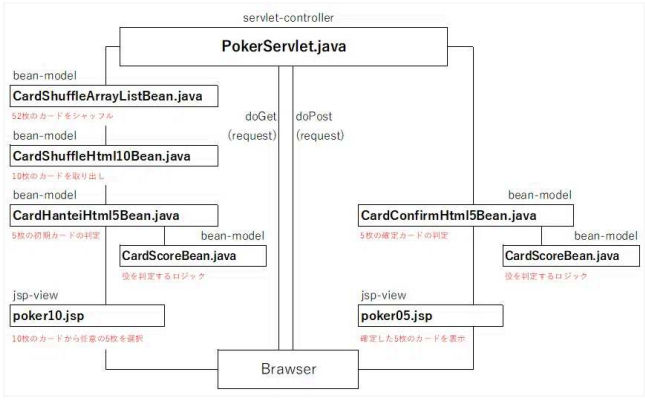

図５．MVCモデル図

カードの処理を行う５つのBeanは提供されます。使い方とクラス図は以下のようになります。

CardShuffleArrayListBean.classの使い方とクラス図

（使い方）
・引数なしのコンストラクタで実体化します。
・shuffleメソッドを実行することでカードの構成がシャッフルされ配列（ArrayList）に保持されます。
・利用側のクラスはgetListメソッドで配列の中身を取得します。

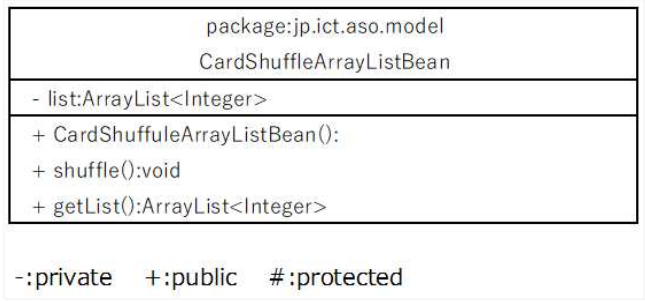

図5.1 CardShuffleArrayListBeanクラス図

CardShuffleHtml10Bean.classの使い方とクラス図

（使い方）
・引数なしのコンストラクタで実体化します。
・setListメソッドにカード構成の配列（ArrayList）を設定すると最初の10枚のカードがhtml文字列でフィールド内に表組されます。この時チェックボックスも付加します。
・最初の5枚は表向きで、後の5枚は裏向きで設定されます。
・利用側のクラスはgetShuffleCardImage10メソッドでフィールドの情報を取得します。

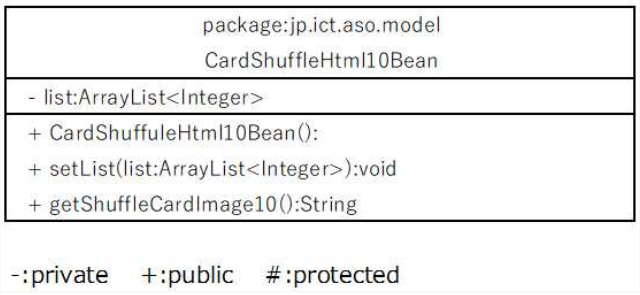

図5.2 CardShuffleHtml10Beanクラス図

CardHanteiHtml5Bean.classの使い方とクラス図

（使い方）
・引数なしのコンストラクタで実体化します。
・setListメソッドに10枚のカード構成の配列（ArrayList）を設定します。
・hanteiCard5メソッドの実行で最初の5枚のカード構成の役を判定します。
・役の判定にはCardScoreBeanクラスを用います。
・利用側のクラスはgetHantei5メソッドで役を取得します。

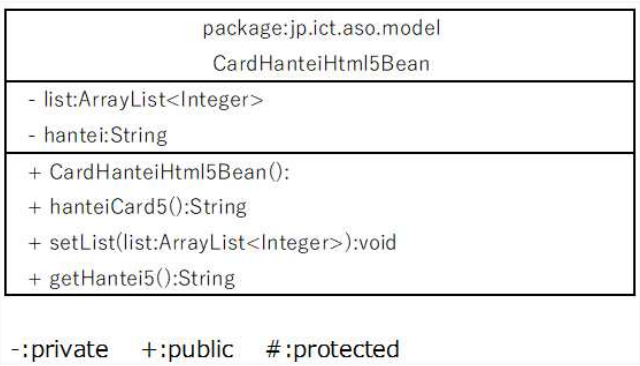

図5.3 CardHanteiHtml5Beanクラス図

CardConfirmHtml5Bean.classの使い方とクラス図

（使い方）
・引数なしのコンストラクタで実体化します。
・setCheckedCardメソッドにチェック済みの5枚のカード構成の配列を設定します。
・setKaketenメソッドで掛け点を設定します。
・hanteiCard5メソッドの実行で5枚のカード構成の役を判定します。
・役の判定と払い戻し点数の計算にはCardScoreBeanクラスを用います。
・利用側のクラスはgetHantei5メソッドで役を取得します。

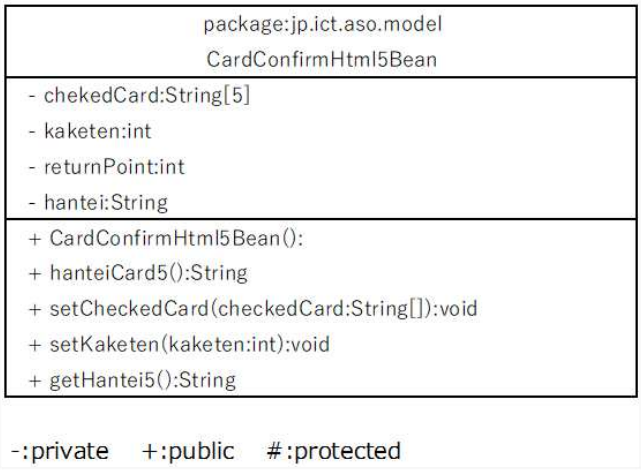

図5.4 CardConfirmHtml5Beanクラス図

CardScoreBean.classの使い方とクラス図

（使い方）
・引数なしのコンストラクタで実体化します。
・setCardメソッドの1～５にカードの数値情報を設定します。
・setKaketenメソッドで掛け点を設定します。
・scoreメソッドの実行で5枚のカード構成の役の判定と払い戻し点数を計算します。
・利用側のクラスはgetMessageメソッドで役を取得し、getReturnPointメソッドで払い戻し点数を取得します。

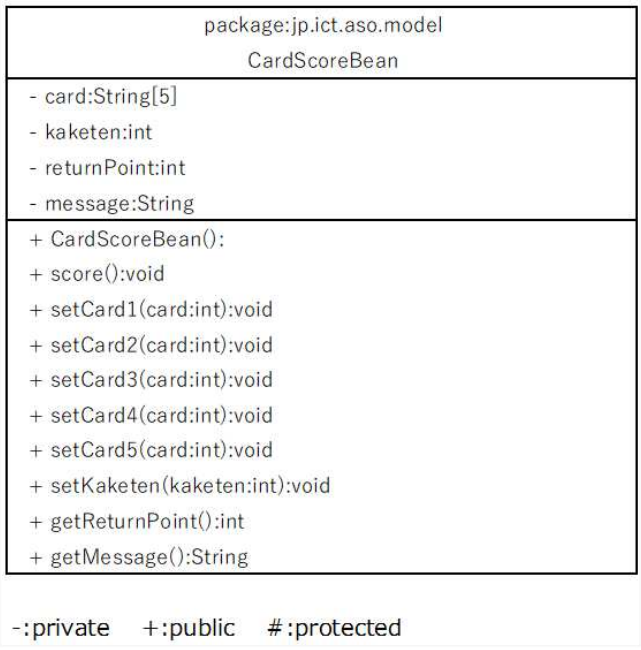

図5.5 CardScoreBeanクラス図

実装準備

１．ポーカーロジック用JavaBeansクラスを使用する準備をします

(1)クラスファイルの準備をします
以下Windows環境を想定しています。
事前に「**提供BeanPokerフォルダ**」を作成しておきます。
（例　c:￥提供BeanPoker）

提供BeanPokerフォルダ内に以下のの**５つのクラスファイル**を保存しておきます。
（CardShuffleArrayListBean.class、CardShuffleHtml10Bean.class、CardHanteiHtml5Bean.class、CardConfirmHtml5Bean.class、CardScoreBean.class）

その際**パッケージの階層**に従ってください。
（例　c:￥提供BeanPoker￥jp￥ict￥aso￥model￥〇〇〇.class）

(2)カード処理を行う５つのクラスファイルをEclipseのビルド・パスに追加します

Eclipseパッケージ・エクスプローラより
gameプロジェクトを右クリック→ビルド・パス→ビルド・パスの構成→「ライブラリ」タブ→「クラスパス」クリック→外部クラス・フォルダーの追加→「提供Bean」を指定します→最後に「適用して閉じる」をクリック

※図６のように一度設定されていれば再度設定する必要はありません。

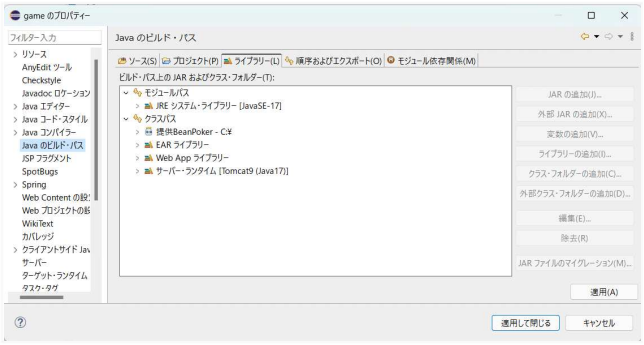

図６．Javaのビルドパス追加画面

２．カードの画像用のフォルダを作成します

Eclipseパッケージ・エクスプローラより
gameプロジェクト → src → main → webapp 内に imagesフォルダを作成し
さらにその中にtrumpフォルダを作成してカードの画像を保存します

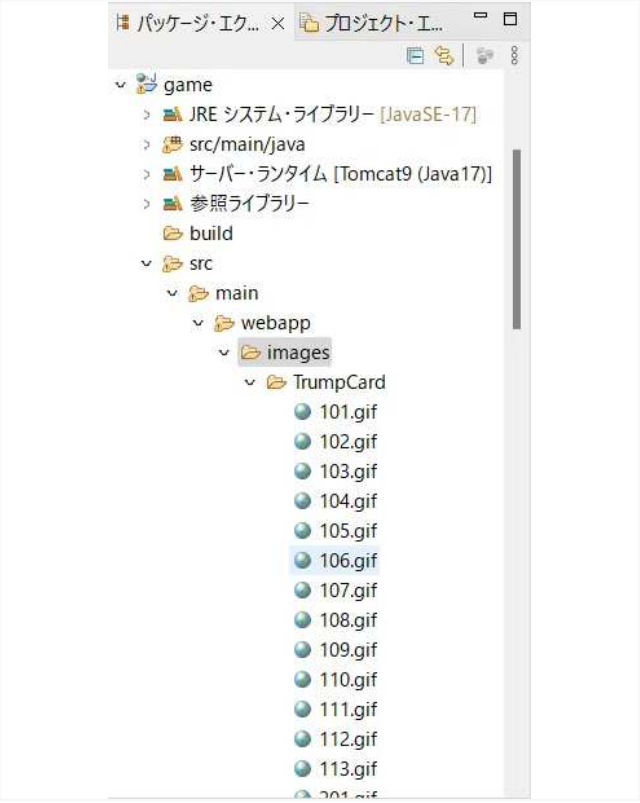

図 6.1 カードの画像の位置

実装手順

１．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
gameプロジェクトを右クリック→新規→その他→Web→サーブレット
→以下の内容で作成する

・PokerServlet.java
　パッケージ：jp.ict.aso.controller
　クラス名：PokerServlet
　ソースコード：考えましょう！　※アノテーションは**/Poker**

【ヒント】

```
doGet
  // カードをシャッフル
  CardShuffleArrayListBean csalb = new CardShuffleArrayListBean();
  csalb.shuffle();
  // シャッフル結果を可変長配列に保存
```

```
        ArrayList list=csalb.getList();
        // カード10枚分を実体化（チェックボックス付き）
        CardShuffleHtml10Bean cs10 = new CardShuffleHtml10Bean();
        cs10.setList(list);
        // １０枚の内最初の５枚の役の判定と表示
        CardHanteiHtml5Bean ch5 = new CardHanteiHtml5Bean();
        ch5.setList(list);
        ch5.hanteiCard5();
        // リクエストスコープに保存
        request.setAttribute("shuffleCard", cs10);
        request.setAttribute("hanteiCard", ch5);

        // poker10.jspにフォワード

    doPost
        // リクエストパラメータの取得
        request.setCharacterEncoding("UTF-8");
        String smochiten = request.getParameter("mochiten");
        String skaketen = request.getParameter("kaketen");
        // 持ち点と掛け点を数値化
        int mochiten = Integer.parseInt(smochiten);
        int kaketen = Integer.parseInt(skaketen);
        // チェック済みのカードをリクエストスコープから取得
        // CardShuffleHtml10Beanでチェック設定されています
        String[] checkedCard = request.getParameterValues("card");
        // 確定の５枚の表示と役の判定
        CardConfirmHtml5Bean cc5 = new CardConfirmHtml5Bean();
        cc5.setCheckedCard(checkedCard);
        cc5.setKaketen(kaketen);
        cc5.hanteiCard5();
        // リクエストスコープに保存
        request.setAttribute("confirmCard", cc5);

        // /poker05.jspにフォワード
```

２．初期配布状態の画面の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
game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

・poker10.jsp
　保存場所：game/src/main/webapp/**WEB-INF/jsp ← 注意！**
　ファイル名：poker10.jsp
　ソースコード：考えましょう！

```
【ヒント】
リクエストスコープから実体化：
  <% CardShuffleHtml10Bean cs10=
        (CardShuffleHtml10Bean)request.getAttribute("shuffleCard");
     CardHanteiHtml5Bean ch5=
        (CardHanteiHtml5Bean)request.getAttribute("hanteiCard"); %>

カード10枚と初期の5枚の役の表示：
    <%=cs10.getShuffleCardImage10() %><br>
    <%=ch5.getHantei5() %><br>
```

３．役確定画面の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
game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

・poker05.jsp
　保存場所：game/src/main/webapp/**WEB-INF/jsp ← 注意！**
　ファイル名：poker05.jsp
　ソースコード：考えましょう！

```
【ヒント】
リクエストスコープから実体化：
  <% CardConfirmHtml5Bean cc5=
      (CardConfirmHtml5Bean)request.getAttribute("confirmCard"); %>

確定のカード5枚と役と配当の表示：
  <%=cc5.getCheckedCardImage5() %><br>
  <%=cc5.getHantei5() %><br>

再配布：
  <a href="Poker" style="text-decoration:none;">[再配布]<br>
  <%=cc5.getCardImage5() %><br>
```

実行確認

サーブレットクラス（PokerServlet.java）を実行します。しかし、、、

１．InternalServerErrorとなります

実行用Beanのデプロイ（配置）が必要です。この設定を行わないと図７のような実行時エラーになります。

Eclipseパッケージ・エクスプローラより
PokerServlet.javaを右クリック→実行→サーバーで実行
→図７のように**エラー**となる

図７．実行時エラー画面

２．実行用Beanを配置します

提供された５つのクラスファイルをEclipseの実行時のクラスパスに追加します。

Eclipse**プロジェクト・エクスプローラ**より
（**パッケージ・エクスプローラではありません！**）
gameプロジェクトを展開→buildを展開→classesを展開→jpを展開→ictを展開→asoを展開→modelがなければ作成

図８のようにmodelフォルダの位置へ提供された５つのクラスファイルをドラッグ＆ドロップすることで実行時クラスパスにBeanが追加されます。
※Eclipseを終了させると、再度追加が必要な場合があります。

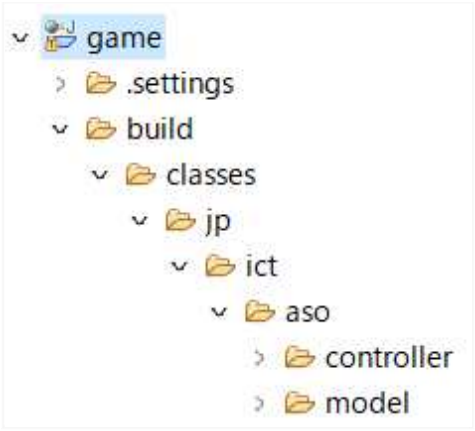

図８．実行時クラスパスの位置

３．再度実行します

Tomcatサーバを再起動したのち、再度サーブレットクラス（PokerServlet.java）を実行します。

Eclipseパッケージ・エクスプローラより
PokerServlet.javaを右クリック→実行→サーバーで実行
→図９のように実行される

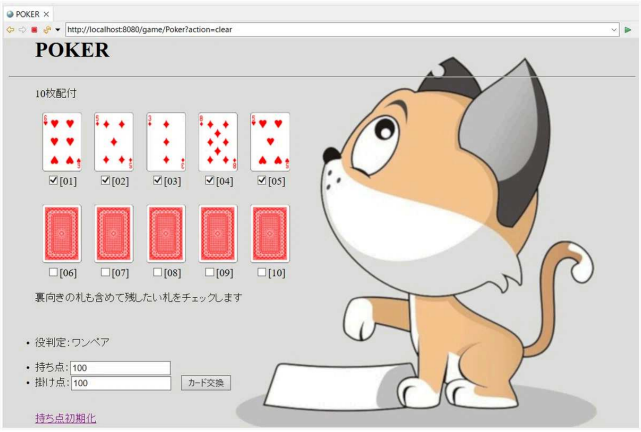

図９．起動画面

テスト

とりあえずPokerのゲーム内で１回のカード交換と役の自動判定が可能になることを確認してください。次に役の判定が問題ないのかテストしなければなりません。このプログラムは**CardScoreBeanクラスで役の判断と配点の計算**を行っています。しかしながら、このクラスには若干のバグがあるようです。**図10（図5.5と同じ）のクラス図**を参考に、CardScoreBeanを実体化するmainメソッドを持った**PokerScoreTest.java**を作成（テスト用スタブを作成）し、すべての役を判定してバグの内容を調査し、問題を報告してください。

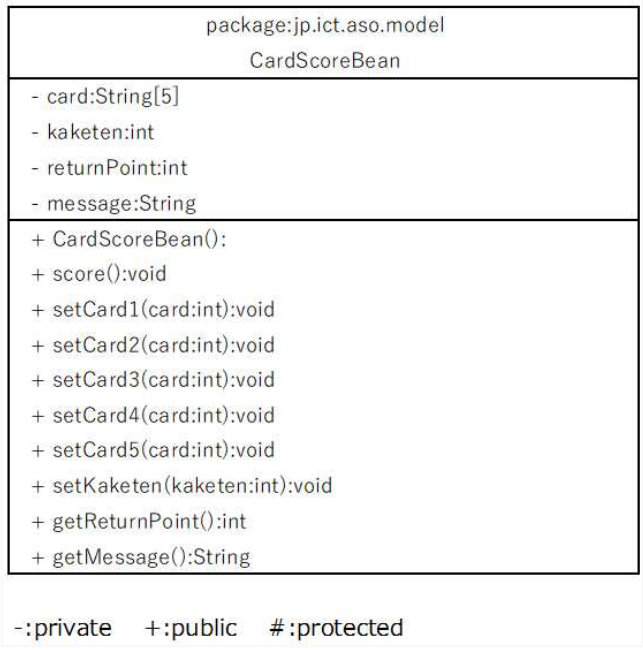

図10. CardScoreBeanクラス図

【ヒント】
図11のように作成するPokerScoreTest.javaは実行可能な（mainメソッドを含んだ）単純なクラスです。CardScoreBeanでは判定するカード５枚の情報をセッターメソッドの引数で与えます。引数で与えるカードの種類は以下のようになり、事前にソートしてセッターメソッドに設定されます。
（カードの種類）
クラブ：101～113　ダイヤ：201～213　ハート：301～313
スペード：401～413

ソースコードの枠組みは以下のようになります。mainメソッド内でCardScoreBeanを実体化して、セッターメソッドに判定するカード番号の５つの組と掛け点を与えます。scoreメソッドで役と配点を計算させてゲッターメソッドで結果を取得し確認してください。

```
package jp.ict.aso.controller;
import jp.ict.aso.model.CardScoreBean;
public class PokerScoreTest {
    public static void main(String[] args) {
    }
}
```

図11. テスト用スタブの枠組み

仕様反映

ここまでできれば、あとは点数計算の結果を反映させないといけません。sessionを使い持ち点、掛け点、配当の計算結果が画面遷移で引き継がれるようにソースコードを変更してください。

ソースコード例と提供Bean