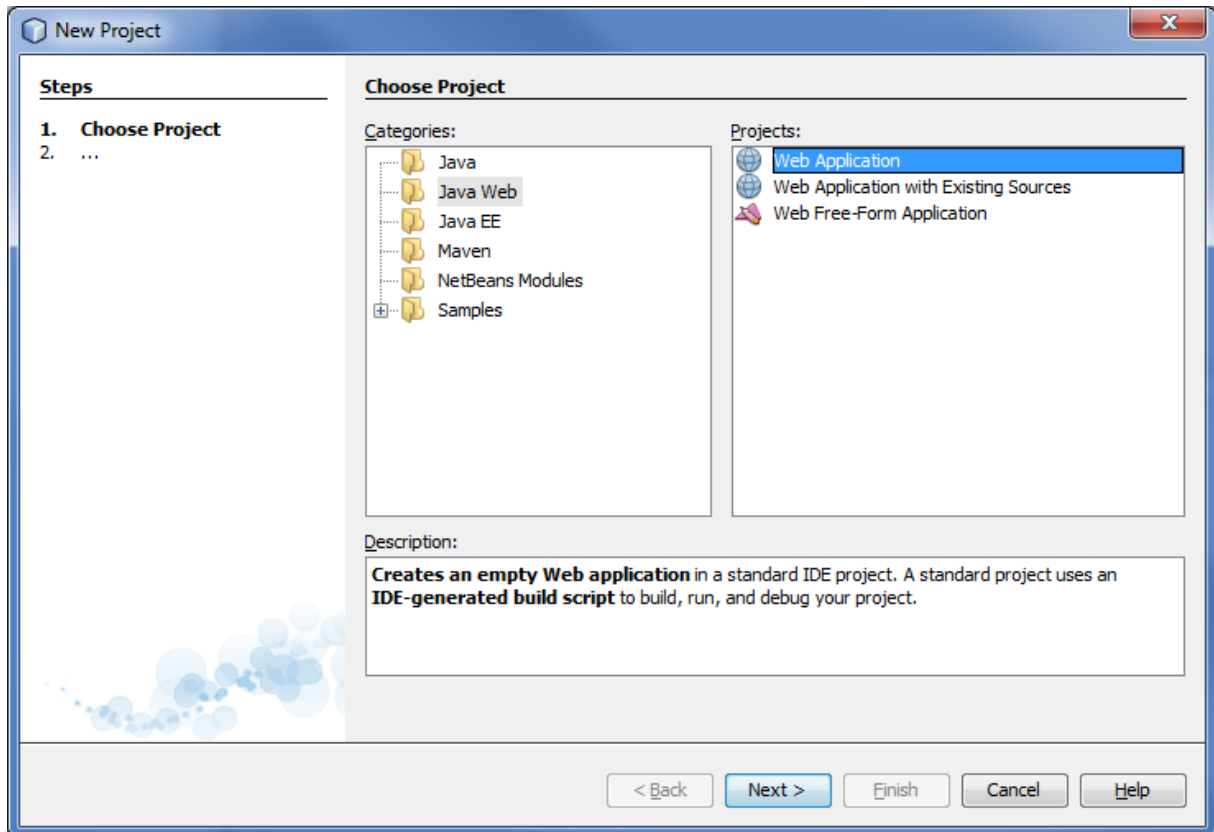


Serwlety, JSP, JSTL

Do realizacji projektu potrzebne jest zintegrowane środowisko programistyczne NetBeans 7 (zrzuty ekranów pochodzą z wersji 7.0.1). Celem ćwiczenia jest wprowadzenie do podstawowych technologii warstwy prezentacji na platformie Java EE, czyli technologii serwletów, JSP i biblioteki znaczników JSTL.

1. Utworzenie projektu webowego

- a) Uruchom środowisko NetBeans. Z menu File wybierz opcję New Project... Wybierz kategorię Java Web i typ projektu Web Application. Kliknij przycisk **Next >**



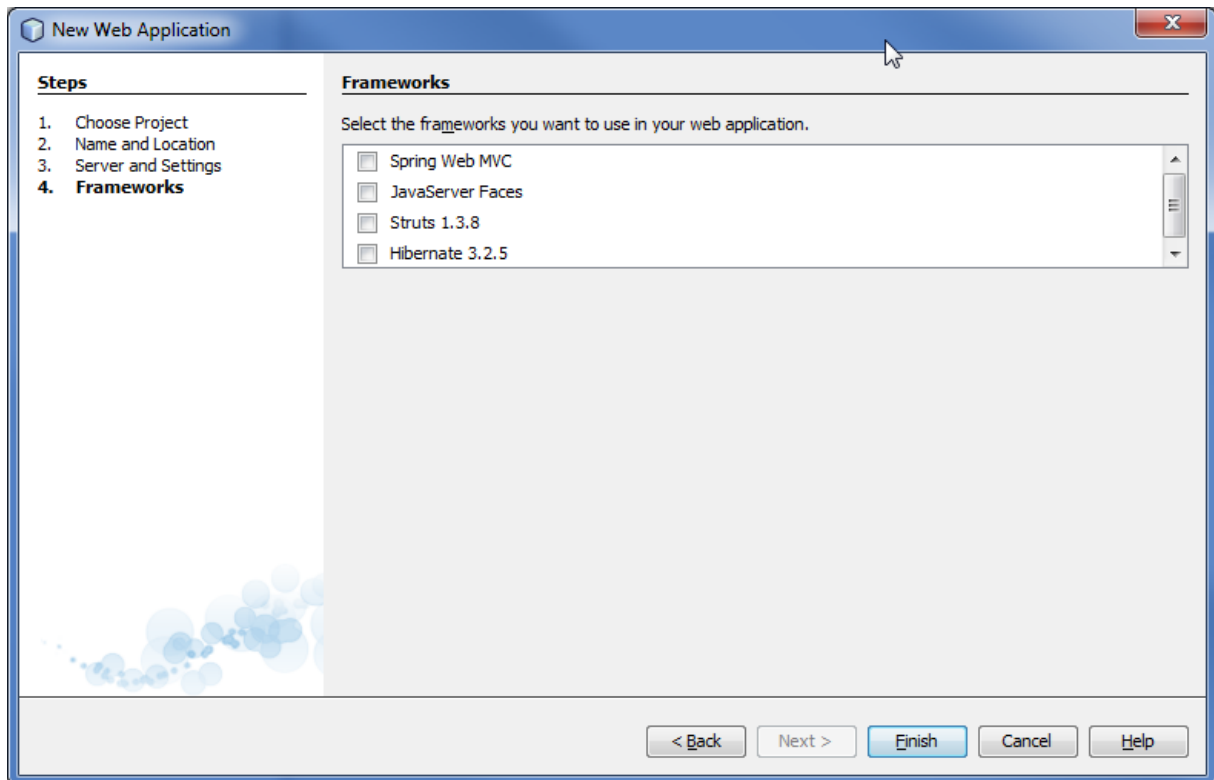
- b) Jako nazwę projektu wpisz **SerwletyJSPJSTL**. Nie zmieniaj wartości pozostałych opcji. Kliknij przycisk **Next >**.

The screenshot shows the 'New Web Application' dialog box with the 'Name and Location' tab selected. The 'Steps' list on the left indicates the current step is '2. Name and Location'. The 'Project Name' field contains 'SerwletyJSPJSTL'. The 'Project Location' field contains 'C:\Workspace\Netbeans' with a 'Browse...' button. The 'Project Folder' field contains 'C:\Workspace\Netbeans\SerwletyJSPJSTL'. There is an unchecked checkbox for 'Use Dedicated Folder for Storing Libraries' and a 'Libraries Folder' field with a 'Browse...' button. Below this, a note states: 'Different users and projects can share the same compilation libraries (see Help for details)'. There is a checked checkbox for 'Set as Main Project'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

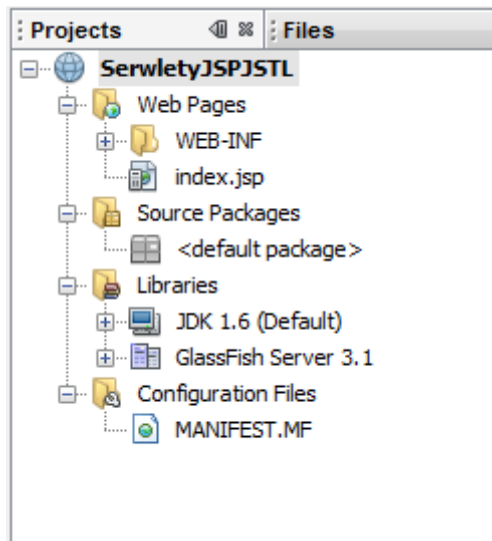
- c) Jako serwer do testowania aplikacji wybierz z listy **GlassFish Server 3.1**. Jako wersję Javy EE wybierz **Java EE 6 Web**. Pozostaw zaproponowaną ścieżkę kontekstu aplikacji (Context Path). Kliknij przycisk **Next >**.

The screenshot shows the 'New Web Application' dialog box with the 'Server and Settings' tab selected. The 'Steps' list on the left indicates the current step is '3. Server and Settings'. The 'Add to Enterprise Application' dropdown is set to '<None>'. The 'Server' dropdown is set to 'GlassFish Server 3.1' with an 'Add...' button. The 'Java EE Version' dropdown is set to 'Java EE 6 Web'. There is an unchecked checkbox for 'Enable Contexts and Dependency Injection'. The 'Context Path' field contains '/SerwletyJSPJSTL'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

- d) W ostatnim kroku kreatora nie zaznaczaj żadnych frameworków. Kliknij przycisk **Finish**.

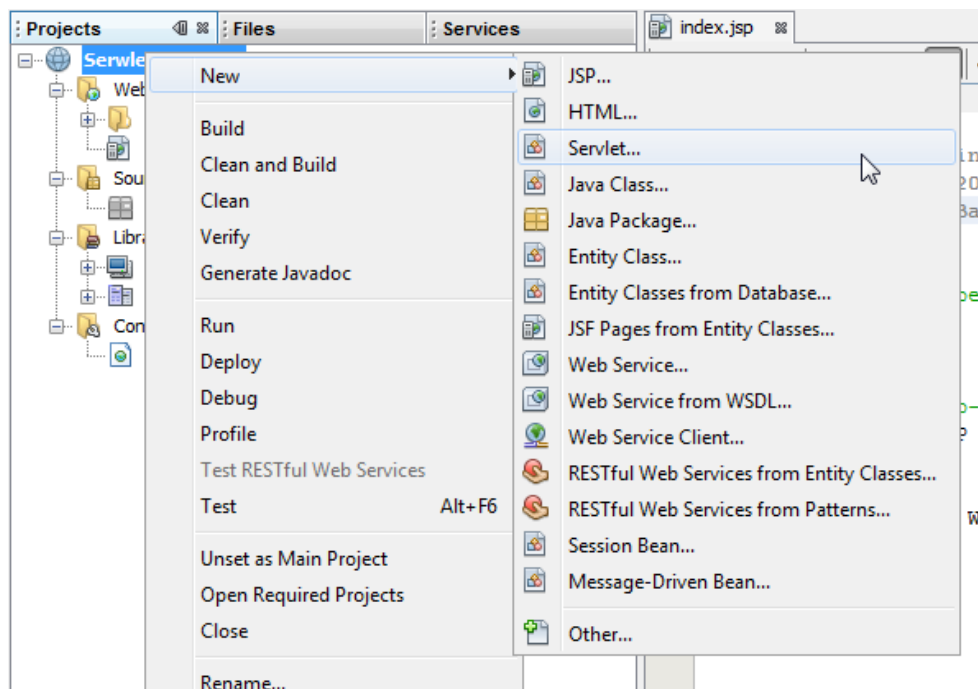


- e) Przeanalizuj strukturę projektu, jaki powstał w wyniku działania kreatora.

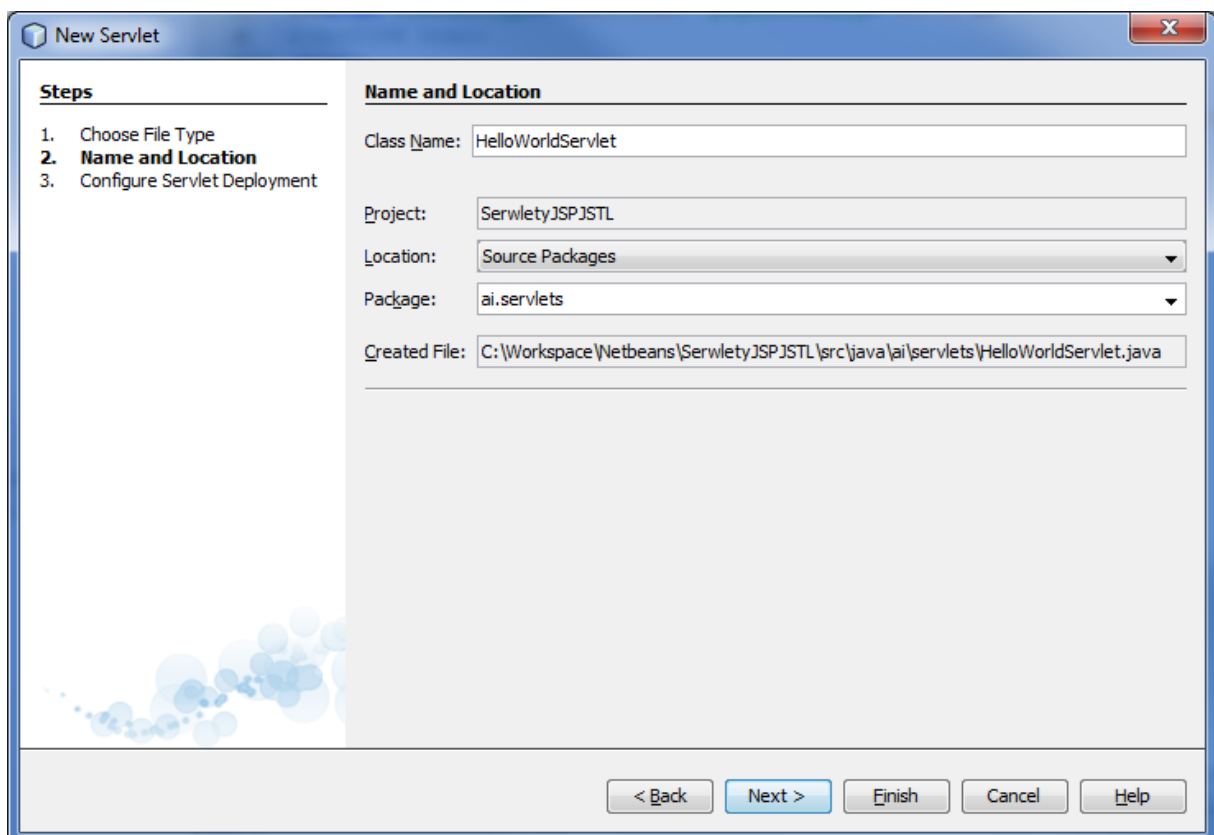


2. Utworzenie i uruchomienie pierwszego serwletu

- a) Kliknij prawym klawiszem na ikonie projektu w nawigаторze projektów, następnie z menu kontekstowego wybierz opcję **New → Servlet**



- b) Jako nazwę serwletu podaj HelloWorldServlet. Jako nazwę pakietu wpisz ai.servlets. Kliknij przycisk **Next >**



- c) W ostatnim kroku kreatora pozostaw odznaczoną opcję dodania informacji o serwlecie do pliku deskryptora instalacji `web.xml`. Zwróć uwagę na to, jaki adres będzie wywoływał serwlet. Kliknij przycisk **Finish**.

Steps

1. Choose File Type
2. Name and Location
3. **Configure Servlet Deployment**

Configure Servlet Deployment

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

☒ Add information to deployment descriptor (`web.xml`)

Class Name:

Servlet Name:

URL Pattern(s):

Initialization Parameters:

Name	Value
------	-------

New Edit... Delete

< Back Next > Finish Cancel Help

Komentarz: Do wersji Java EE 5 konfiguracja serwletów (i całego modułu webowego aplikacji Java EE) realizowana była w oparciu o plik `web.xml`. Od wersji Java EE 6 plik ten jest opcjonalny, gdyż podstawowa konfiguracja może być wyspecyfikowana poprzez adnotacje w kodzie Java. Plik `web.xml` jest używany do „nadpisania” ustawień zawartych w aplikacji po jej instalacji (bez konieczności rekompilacji aplikacji) oraz dla ustawień zaawansowanych, których nie można zrealizować poprzez adnotacje.

- d) Po zakończeniu działania kreatora kod źródłowy serwletu zostanie otwarty w edytorze kodu. Odszukaj adnotację specyfikującą nazwę serwletu i jego odwzorowanie (`@WebServlet`). Przeanalizuj w jaki sposób kreator zapewnił, że serwlet będzie obsługiwał żądania GET i POST, reagując na nie w ten sam sposób.

- e) Usuń z klasy serwletu metody `doGet`, `doPost`, `processRequest` i `getServletInfo`. Następnie dodaj poniższą implementację metody `doGet`. Wygeneruje ona dynamiczną stronę WWW, zatytułowaną „Hello World Servlet”, wyświetlającą w przeglądarce tekst „Hello World”.

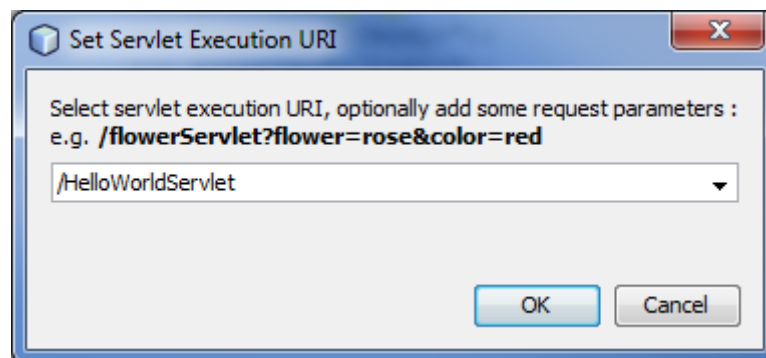
```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

    response.setContentType("text/html");
    response.setCharacterEncoding("windows-1250");

    PrintWriter out = response.getWriter();

    out.println("<html>");
    out.println("<head><title>Hello World Servlet</title></head>");
    out.println("<body>");
    out.println("<h1>Hello World!</h1>");
    out.println("</body>");
    out.println("</html>");
    out.close();
}
```

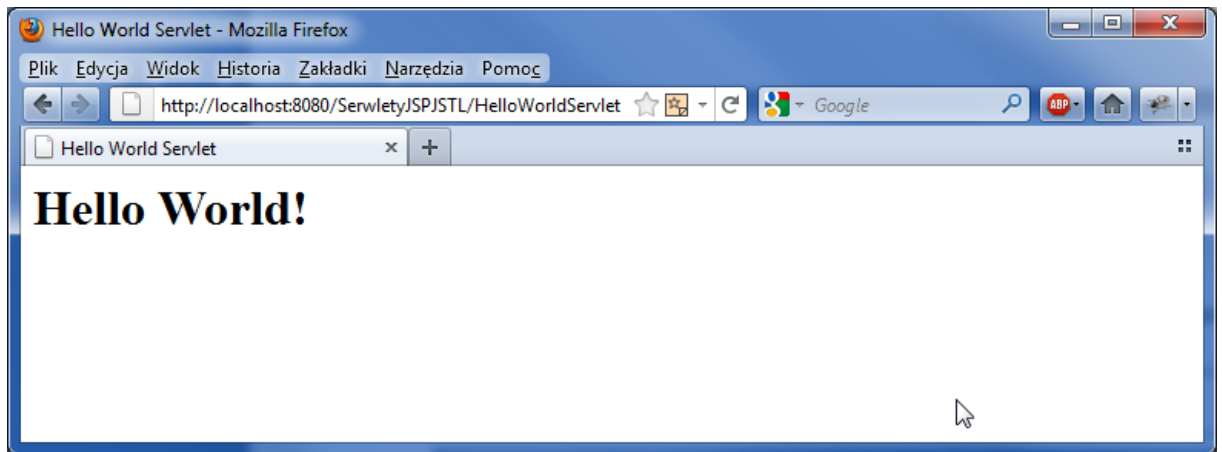
- f) Uruchom serwlet za pomocą opcji **Run File** z menu kontekstowego serwletu (z poziomu nawigatora projektów lub edytora, nie używaj opcji **Run Main Project** – uruchomi ona plik `index.jsp`). NetBeans zaproponuje dodanie parametrów wywołania serwletu. Nie zmieniaj adresu URI i kliknij przycisk **OK**.



- g) Zaobserwuj komunikaty wyświetlane na konsoli serwera świadczące o tym, że w celu przetestowania serwletu został uruchomiony serwer aplikacji GlassFish i została na nim zainstalowana aplikacja zawierająca serwlet.



- h) Zwróć uwagę na adres wywołujący serwlet w pasku adresu przeglądarki internetowej (NetBeans powinien ją automatycznie uruchomić, kierując ją do właściwego adresu).

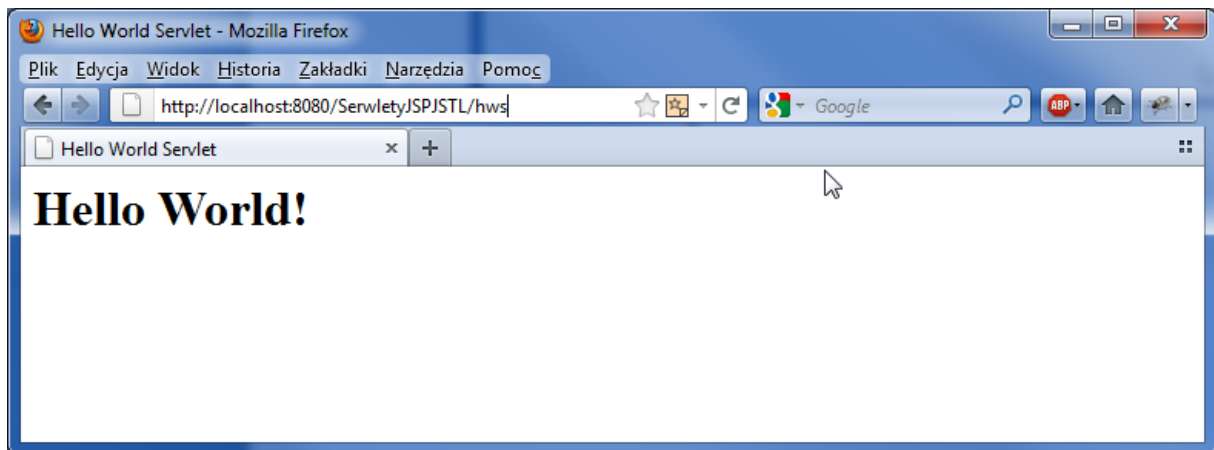


3. Zmiana odwzorowania serwletu na adres URL

- a) Wróć do edycji kodu klasy serwletu. Poprzez modyfikację adnotacji zmień domyślne odwzorowanie na /hws.

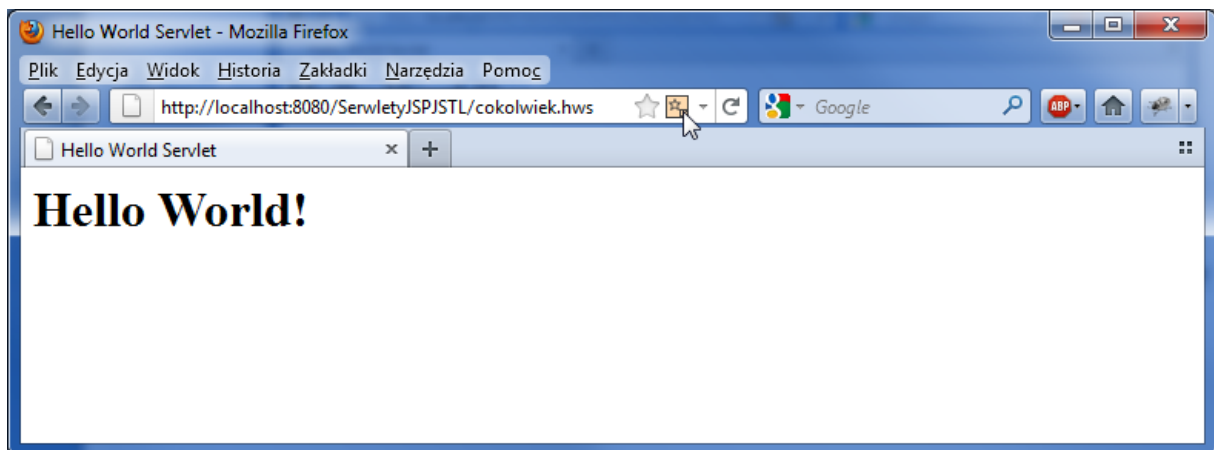
```
@WebServlet(name = "HelloWorldServlet", urlPatterns = {"/hws"})
```

Ponownie zainstaluj aplikację na serwerze, tym razem bez automatycznego uruchamiania przeglądarki przez NetBeans. W tym celu w oknie Projects z menu kontekstowego dla projektu wybierz opcję Deploy. Wróć do przeglądarki i odśwież zawartość dokumentu (nie zmieniaj adresu). Co zauważyłaś/eś? Następnie uruchom w przeglądarce adres `http://localhost:8080/SerwletyJSPJSTL/hws`

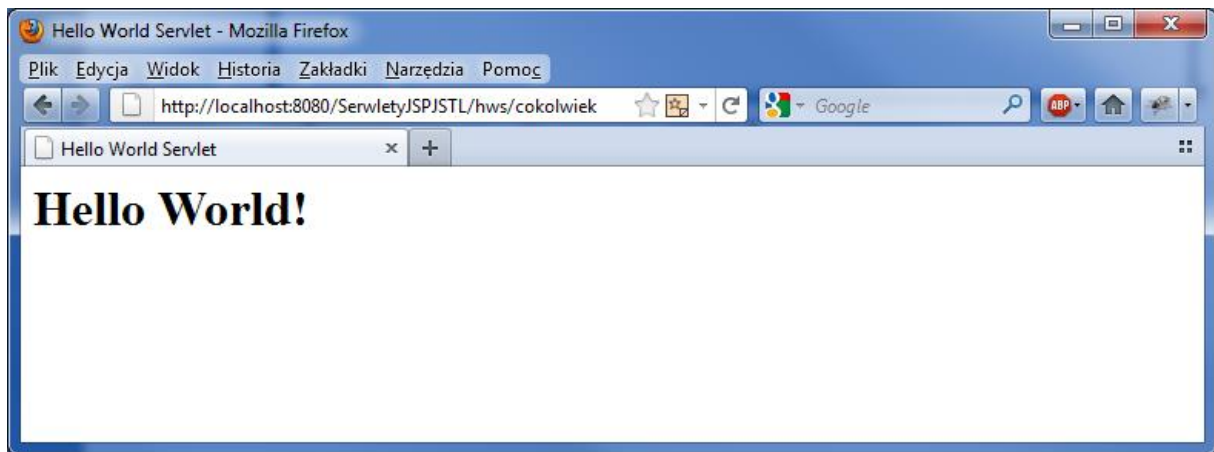


- b) Powtórz kroki z punktu a) dla następujących dwóch odwzorowań: `*.hws`, `/hws/*`. Chcemy uzyskać poniższe adresy wywołania serwletu:

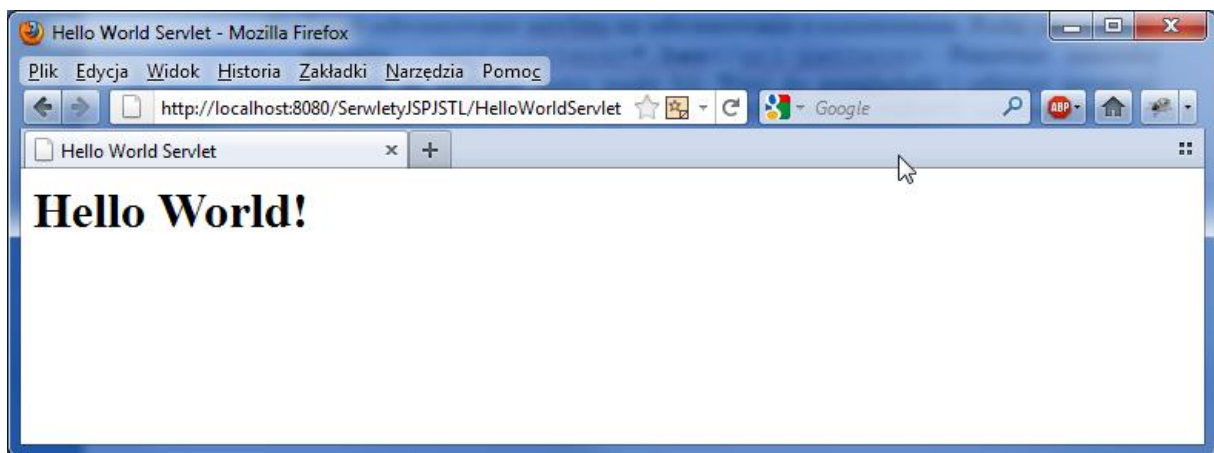
- `http://localhost:8080/SerwletyJSPJSTL/cokolwiek.hws`



- `http://localhost:8080/SerwletyJSPJSTL/hws/cokolwiek`



c) Przywróć pierwotne odwzorowanie serwletu.



4. Przekazywanie parametrów z formularza HTML do serwletu

- a) Kliknij prawym klawiszem myszy na ikonie projektu i wybierz New → HTML. Jako nazwę dokumentu podaj form (rozszerzenie *.html zostanie dodane automatycznie). Umieść w dokumencie poniższy kod.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
  <head>
    <title>HTML form</title>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
  </head>
  <body>
    <form action="HelloWorldServlet" method="get">
      <table>
        <tr><td>Name:</td><td><input type="text" name="name"/></td></tr>
        <tr><td>Age: </td><td><input type="text" name="age" size="2"/></td></tr>
        <tr><td colspan="2"><input type="submit" value="Send"/></td></tr>
      </table>
    </form>
  </body>
</html>
```

- b) W edytorze kodu otwórz zawartość pliku HelloWorldServlet.java. Zmodyfikuj kod metody doGet() w poniższy sposób. Zwróć uwagę na konieczność jawnego rzutowania parametrów innych niż łańcuchy znaków.

```
response.setContentType("text/html");
response.setCharacterEncoding("windows-1250");

PrintWriter out = response.getWriter();

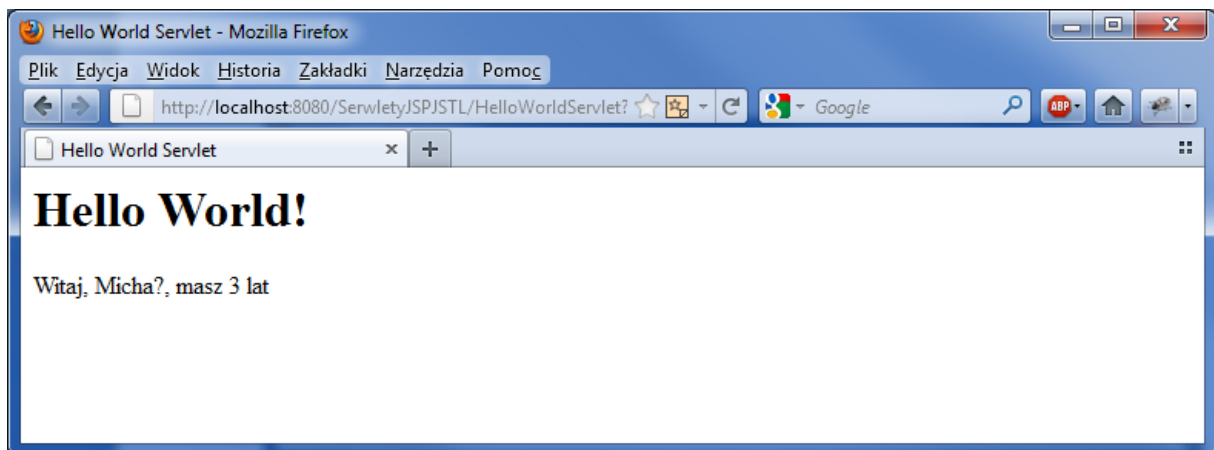
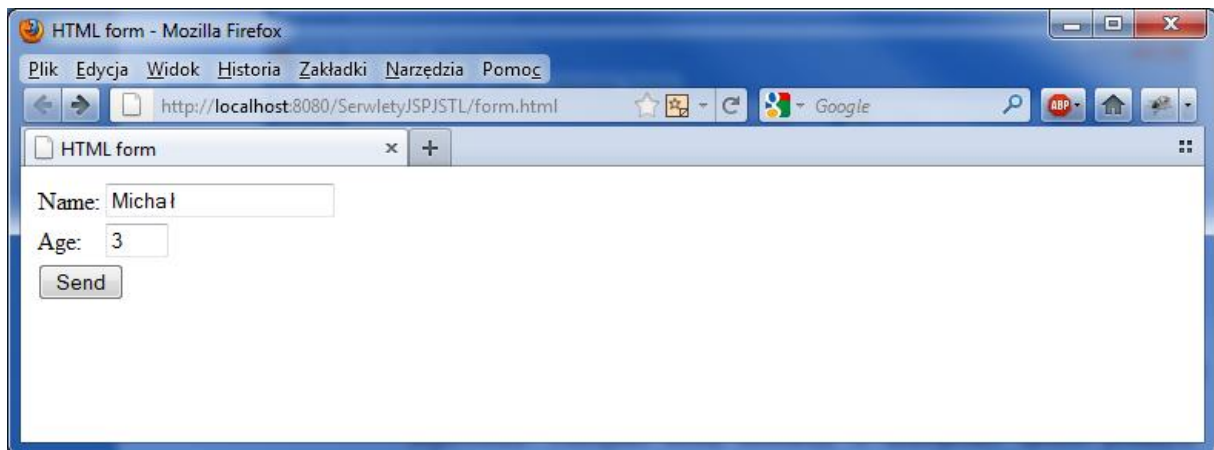
String name = request.getParameter("name");
int age = Integer.parseInt(request.getParameter("age"));

out.println("<html>");
out.println("<head><title>Hello World Servlet</title></head>");
out.println("<body>");
out.println("<h1>Hello World!</h1>");

out.println("<p>Witaj, " + name + ", masz " + age + " lat</p>");

out.println("</body>");
out.println("</html>");
out.close();
```

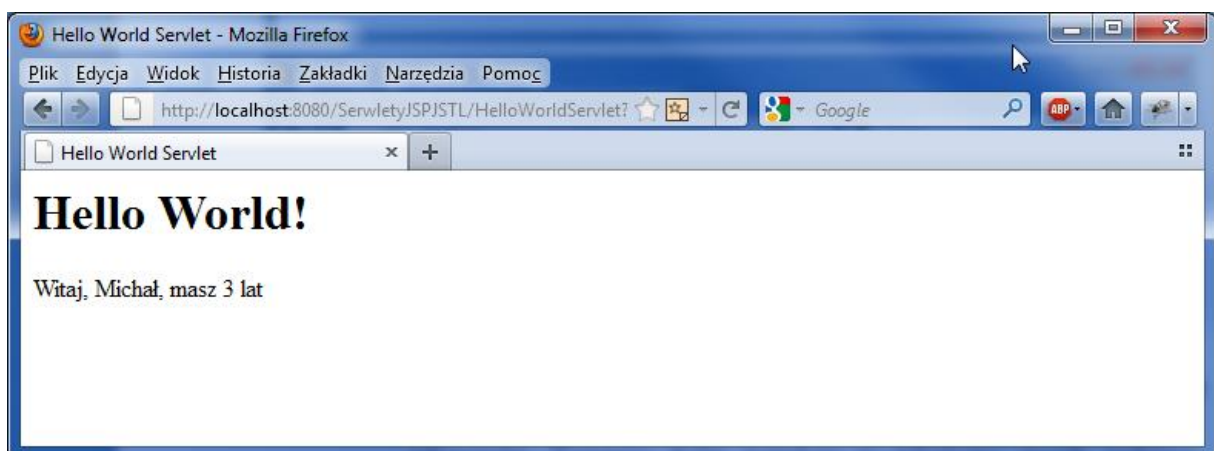
- c) Kliknij prawym klawiszem myszy na pliku form.html (w oknie Projects, gałąź Web Pages), z menu kontekstowego wybierz pozycję Run File. Wypełnij pola formularza i prześlij formularz do przetworzenia w serwlecie.



- d) Odszukaj w kodzie serwletu wiersz ustawiający kodowanie dla generowanej odpowiedzi. Następnie dodaj instrukcję w anlogiczny sposób podającą informację o kodowaniu dla żądania (obiekt request). Zwróć uwagę aby informacja ta była ustawiona przed odczytem wartości parametrów z żądania.

```
request.setCharacterEncoding("windows-1250");
```

Ponownie uruchom formularz i sprawdź działanie serwletu dla imienia z polskimi znakami.



- e) Zmień w formularzu HTML metodę wywoływania serwletu na POST.

```
<form action="HelloWorldServlet" method="post">
```

Zainstaluj aplikację na serwerze. Ponownie otwórz formularz w przeglądarce i spróbuj wysłać formularz do serwletu. Zinterpretuj otrzymany komunikat.

- f) Do klasy `HelloWorldServlet` dodaj metodę `doPost`, która będzie obsługiwała metodę wywołania POST. Następnie zainstaluj aplikację na serwerze i ponownie spróbuj przesłać wypełniony formularz do serwletu.

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    this.doGet(request, response);
}
```

5. Obsługa sesji w serwlecie

- a) Utwórz w projekcie nowy serwlet i nazwij go `HttpSessionServlet`. Umieść nowo utworzony serwlet w pakiecie `ai.servlets`. Serwlet będzie umożliwiał przechowywanie krótkich notatek pomiędzy kolejnymi wywołaniami dokumentu. Pamiętaj, aby nie zaznaczać opcji dodania informacji o serwlecie do pliku deskryptora instalacji `web.xml`.
- b) Jako kod serwletu podaj poniższy kod. Zwróć szczególną uwagę na obsługę sesji, utworzenie nowego obiektu do przechowywania notatek w przypadku gdy obiekt sesyjny jest pusty oraz konieczność każdorazowego utrwalenia obiektu sesyjnego. Uruchom serwlet (użyj polecenia `Run File` dla serwletu) i przetestuj jego działanie. Przetestuj działanie serwletu dla dwóch współbieżnych sesji (uruchom inną przeglądarkę i skopiuj adres wywołujący serwlet do paska adresu).

```
package ai.servlets;

import java.io.*;
import java.util.ArrayList;
import javax.servlet.annotation.WebServlet;
import javax.servlet.*;
import javax.servlet.http.*;

@WebServlet(name="HttpSessionServlet", urlPatterns={"/HttpSessionServlet"})
public class HttpSessionServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

        HttpSession session = request.getSession();

        ArrayList notes = (ArrayList)session.getAttribute("notes");
        if (notes == null) {
            notes = new ArrayList ();
            session.setAttribute("notes",notes);
        }

        String note = request.getParameter("note");
        if (note != null)
            notes.add(note);

        PrintWriter out = response.getWriter();

        out.println("<html>"); out.println("<body>");
        out.println("<h1>My notes</h1>");
        out.println("<ul>");

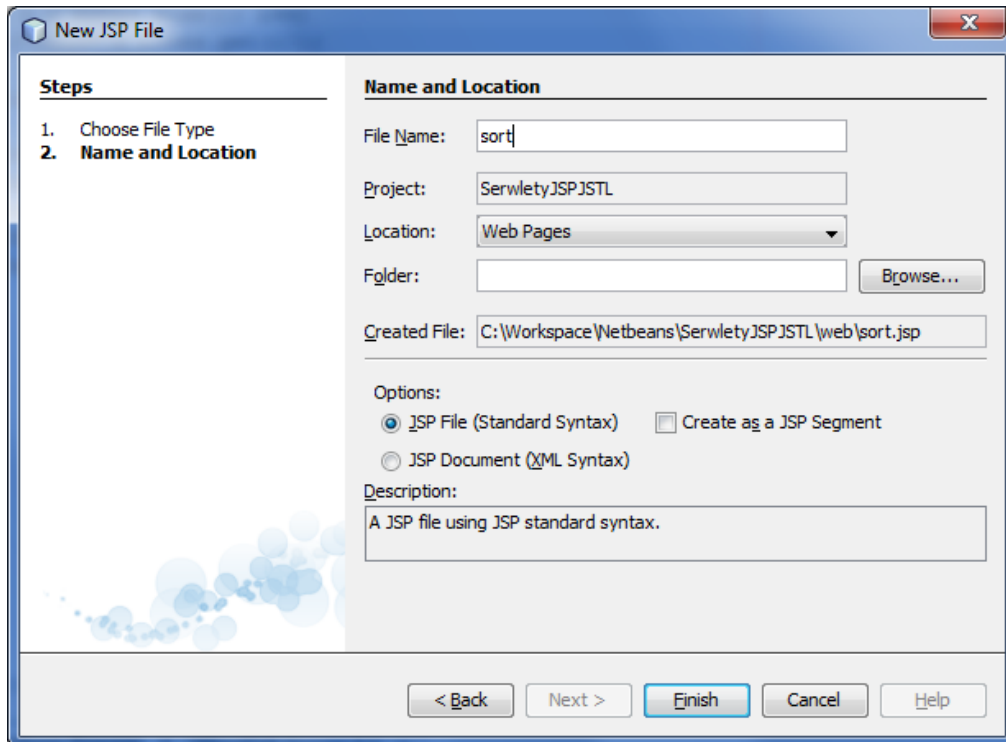
        for (int i = 0; i < notes.size(); i++)
            out.println("<li>" + notes.get(i));

        out.println("</ul>");

        out.println("<form action='HttpSessionServlet'>");
        out.println("<input type='text' name='note'><br/>");
        out.println("<input type='submit' value='Add note'>");
        out.println("</form>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

6. Utworzenie w projekcie nowej strony JSP

- a) Kliknij prawym przyciskiem myszy na węzeł **Web Pages** w drzewie projektu. Wybierz z menu kontekstowego polecenie **New** → **JSP...**. Podaj nazwę pliku JSP: `sort`. Uwaga! Nie podawaj rozszerzenia pliku gdyż zostanie ono dodane automatycznie. Kliknij przycisk **Finish**.



- b) Przejdź do edycji pliku `sort.jsp`. Strona, generowana przez plik `sort.jsp`, będzie realizowała sortowanie bąbelkowe zbioru liczb wyspecyfikowanych w definicji tej strony. Wprowadź do pliku `sort.jsp` następujący kod.

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Bubble Sort</title>
  </head>
  <body>
    <%! int temp, i; %>
    <%! boolean change; %>
    <%! int data[] = { 1, 6, 4, 5, 3 }; %>
    <%! int size; %>

    <% size = data.length;
    do {
      change = false;
      i = size - 1;
      do {
        i--;
        if (data[i+1] < data[i]) {
          temp = data[i];
          data[i] = data[i+1];
          data[i+1] = temp;
        }
      } while (i > 0);
      change = !change;
    } while (change);
    %>
```

```
        data[i+1] = temp;
        change = true;
    }
    } while (i != 0);
} while (change); %>
Posortowane liczby:
<% for (int i = 0; i < size; i++) { %>
    <%= data [i] %>
    <% } %>
</body>
</html>
```

- c) Uruchom stronę za pomocą opcji Run File z menu kontekstowego wyświetlanego po kliknięciu prawym klawiszem myszy na nazwę pliku `sort.jsp` w oknie projektu. Uruchomienie strony oznacza oczywiście uruchomienie serwera aplikacji i zainstalowanie na nim aplikacji zawierającej testowaną stronę JSP. Komunikat o uruchomieniu serwera zostanie wyświetlony u dołu ekranu.

7. Strona JSP z formularzem.

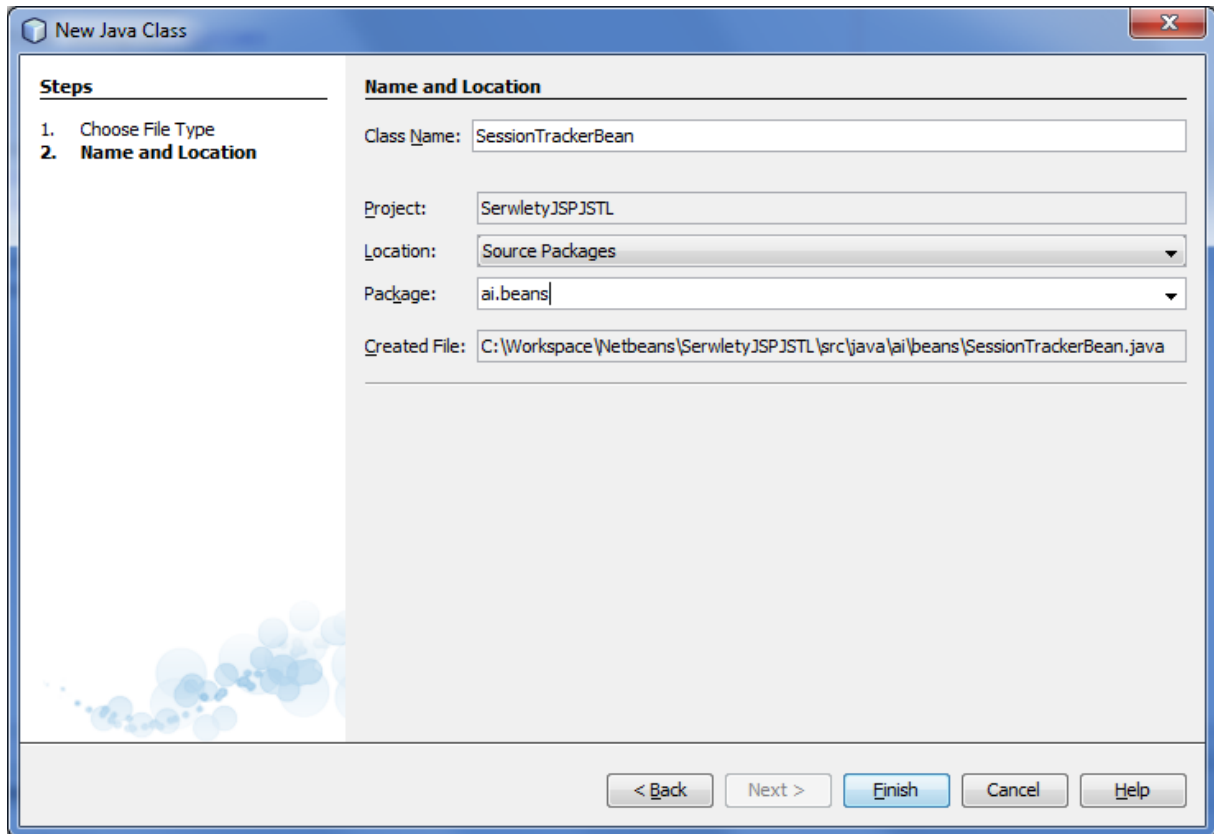
Jeśli z jakimś formularzem nie jest związany atrybut `action`, wówczas zawartość formularza zostaje ponownie wysłana do tej samej strony JSP. Utwórz w projekcie nową stronę JSP `form.jsp`, umieść w niej poniższy kod a następnie przetestuj działanie strony.

```
<%@page contentType="text/html; charset=windows-1250"%>
<%@page pageEncoding="windows-1250"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <title>Form example</title>
    </head>
    <body>
        <form>
            Nazwisko: <input type="text" name="last_name"/>
            <input type="submit" value="wyślij"/>
        </form><br/>
        <% request.setCharacterEncoding("windows-1250"); %>
        <% if (request.getParameter("last_name") != null) { %>
            Przesłane nazwisko to <%= request.getParameter("last_name") %>
        <% } %>
    </body>
</html>
```


8. Wykorzystanie komponentów Java Bean do odseparowania kodu wykonywalnego na stronach JSP
- a) Kliknij prawym klawiszem myszy na ikonie projektu i z menu kontekstowego wybierz **New** → **Java Class**. Utwórz klasę `SessionTrackerBean` i umieść ją w pakiecie `ai.beans`.



- b) Umieść w klasie następujący kod. Zanalizuj strukturę definiowanej klasy.

```
package ai.beans;

import java.util.Date;

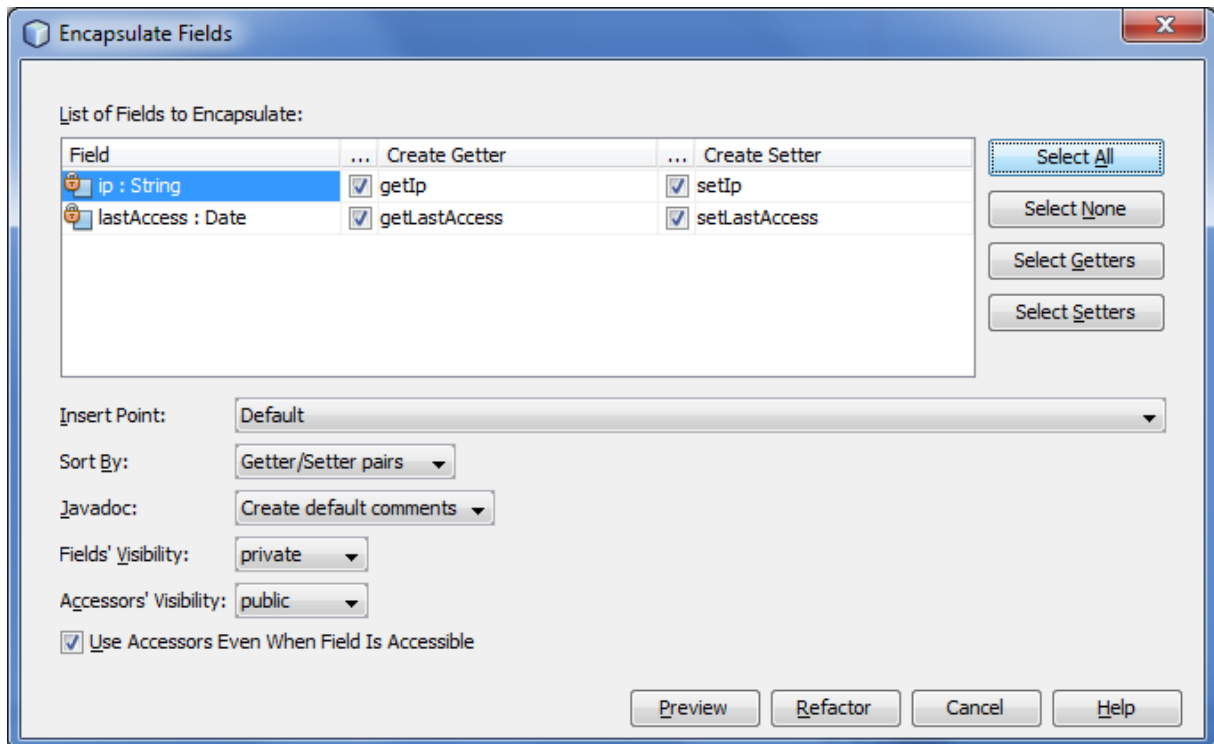
public class SessionTrackerBean {

    private String ip;
    private Date lastAccess;

    public SessionTrackerBean() {
    }

}
```

- c) Wygeneruj zestaw metod obsługujących składowe klasy. W tym celu kliknij prawym klawiszem myszy w edytorze kodu i z menu kontekstowego wybierz opcję **Refactor** → **Encapsulate Fields**. Upewnij się, że dla obu składowych zostaną wygenerowane metody *getter* i *setter*. Kliknij przycisk **Refactor**.



- d) Utwórz nową stronę JSP o nazwie `sessionBean.jsp`. Umieść w niej poniższy kod. Zwróć uwagę na sposób wykorzystania komponentu Java Bean: inicjalizacji, odczytania i zapisania własności komponentu. Uruchom stronę i kilkakrotnie ją odśwież.

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@page import="java.util.Date" %>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
<head><title>JSP Beans</title></head>
<body>
    <jsp:useBean id="myBean" class="ai.beans.SessionTrackerBean" scope="session"/>

    <h2>JSP Session Bean</h2>

    Dane odczytane z komponentu Java Bean:
    <ul>
        <li>adres IP: <jsp:getProperty name="myBean" property="ip"/>
        <li>ostatni dostęp: <jsp:getProperty name="myBean" property="lastAccess"/>
    </ul>

    <jsp:setProperty name="myBean" property="ip"
        value="<%= request.getRemoteAddr() %>"/>
    <jsp:setProperty name="myBean" property="lastAccess"
        value="<%= new Date() %>"/>

</body>
</html>
```

9. Język wyrażeń JSP (JSP EL) jako podstawowe obecnie narzędzie dostępu do zmiennych, parametrów i nagłówków.

Utwórz nową stronę JSP o nazwie `jspEL.jsp`, wypełnij ją poniższym kodem a następnie uruchom. Zwróć uwagę na różne sposoby odczytywania zmiennych za pomocą języka wyrażeń.

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@page import = "java.util.Date" session="true"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head><title>JSP EL Test</title></head>
    <body>
        <form method="post">
            Podaj imię: <input type="text" name="username" />
            <input type="submit" value="Prześlij" />
        </form><br/>

        <h2>Język wyrażeń JSP EL</h2>

        Imię: ${param.username} <br/>
        Przeglądarka: ${header["user-agent"]} <br/>
        Ostatnia wizyta: ${sessionScope.lastVisit} <br/>

        <% session.setAttribute("lastVisit", new java.util.Date()); %>
    </body>
</html>
```

10. Współpraca między formularzem, serwletem, komponentem Java Bean oraz stroną JSP. (przekierowanie żądania z serwletu do JSP)

- a) Utwórz w projekcie prostą stronę html o nazwie yesterday.html a następnie wypełnij ją poniższym kodem.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
  <head>
    <meta http-equiv="Content-Type"
          content="text/html; charset=windows-1250"/>
    <title>Data entry form</title>
  </head>
  <body>
    <form action="ControllerServlet" method="post">
      <table>
        <tr><td>tekst</td>
          <td><select name="foreColor">
            <option>red</option><option>green</option><option>black</option>
          </select></td></tr>
        <tr><td>tło</td>
          <td><select name="backColor">
            <option>yellow</option><option>magenta</option><option>gray</option>
          </select></td></tr>
        <tr><td colspan="2"><input type="submit" value="wyślij"></td></tr>
      </table>
    </form>
  </body>
</html>
```

- b) Przygotuj klasę Java Bean, która będzie wykorzystywana do komunikacji między serwletem i stroną JSP. Dodaj do projektu nową klasę ColorBean (w pliku ColorBean.java), umieść klasę w pakiecie ai.beans. Zastąp kod wygenerowany automatycznie poniższym. Nie zapomnij wygenerować metod *getter* i *setter* i dla obu pól składowych (użyj mechanizmu refaktoryzacji kodu).

```
package ai.beans;

public class ColorBean {

    private String foregroundColor;
    private String backgroundColor;

    public ColorBean() {
    }
}
```

- c) W kolejnym kroku dodaj do projektu serwlet o nazwie `ControllerServlet` i umieść go w pakiecie `ai.servlets` (serwlet jest wykorzystany w metodzie `action` formularza z pliku `yesterday.html`). Kod serwletu znajduje się poniżej. Zwróć uwagę na wykorzystanie komponentu Java Bean do komunikacji między warstwami aplikacji.

```
package ai.servlets;

import java.io.*;

import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.WebServlet;

import ai.beans.ColorBean;

@WebServlet(name = "ControllerServlet", urlPatterns = {"/ControllerServlet"})
public class ControllerServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {
        doPost(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {
        ColorBean myBean = new ColorBean();
        myBean.setForegroundColor( request.getParameter("foreColor") );
        myBean.setBackgroundColor( request.getParameter("backColor") );
        request.setAttribute("bean", myBean);

        ServletContext ctx = this.getServletContext();
        RequestDispatcher dispatcher =
            ctx.getRequestDispatcher("/yesterday.jsp");
        dispatcher.forward(request, response);
    }
}
```

- d) W ostatnim kroku stwórz stronę JSP o nazwie yesterday.jsp i wypełnij poniższym kodem. Następnie wróć do strony yesterday.html i uruchom aplikację. Przetestuj jej działanie.

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

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<html>
<head>
    <title>Yesterday</title>
    <style type="text/css">
        body { background: ${bean.backgroundColor};
              color: ${bean.foregroundColor} }
    </style>
</head>
<body>

<h2>Yesterday</h2>

    Yesterday,<br/>
    All my troubles seemed so far away,<br/>
    Now it looks as though they're here to stay,<br/>
    Oh, I believe in yesterday.<br/>
    <br/>
    Suddenly,<br/>
    I'm not half the man I used to be,<br/>
    There's a shadow hanging over me,<br/>
    Oh, yesterday came suddenly.<br/>
    <br/>
    Why she<br/>
    Had to go I don't know, she wouldn't say.<br/>
    I said,<br/>
    Something wrong, now I long for yesterday.<br/>

    <a href="yesterday.html">powrót</a>
</body>
</html>
```

11. JSTL – Standardowa biblioteka znaczników dla JSP

- a) W kodzie serwletu z poprzedniego zadania umieść poniższe instrukcje (przed przekierowaniem żądania do JSP) rejestrujące w zasięgu bieżącego żądania 2 obiekty: datę wydania piosenki i kolekcję ze składem zespołu:

```
ArrayList members = new ArrayList();
members.add("John Lennon");
members.add("Paul McCartney");
members.add("Ringo Starr");
members.add("George Harrison");
request.setAttribute("members", members);

Calendar calDate = new GregorianCalendar();
calDate.set(1965, Calendar.SEPTEMBER, 13);
Date releaseDate = calDate.getTime();
request.setAttribute("releaseDate", releaseDate);
```

- b) Zaimportuj wykorzystywane we wklejonym kodzie klasy ArrayList, Date, Calendar i GregorianCalendar z pakietu java.util.
- c) W kodzie strony JSP, wyświetlającej dane przygotowane przez serwlet (yesterday.jsp), dodaj (pod tekstem piosenki) poniższy kod:

```
<p>
Released: <c:out value="${releaseDate}" />
</p>
<c:if test="${!empty members}">
  <table border>
    <tr><th>The Beatles</th></tr>
    <c:forEach var="member" items="${members}">
      <tr><td>${member}</td></tr>
    </c:forEach>
  </table>
</c:if>
```

- d) Nie zapomnij dodać (jako pierwszego wiersza kodu strony JSP) dyrektywy deklarującej użycie wykorzystywanego zestawu znaczników JSTL:

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
```

- e) Uruchom i przetestuj aplikację (uruchamiając formularz HTML)
- f) Zastąp znacznik wyświetlający przekazaną datę znacznikiem formatującym:

```
<fmt:formatDate value="${releaseDate}" type="date" dateStyle="default"/>
```

- g) Nie zapomnij o dyrektywie deklarującej użycie zestawu znaczników formatujących JSTL:

```
<%@taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- e) Przetestuj inne formaty daty: short, medium, long i full (zastępuj wartość default dla atrybutu `dateStyle`).