Budowa aplikacji wielowarstwowych. Zastosowanie szablonów, tabel oraz plików typu properties

Laboratorium 3

Technologie internetowe

Zofia Kruczkiewicz

Wykaz pytań dotyczących materiału wykorzystanego w lab3, które należy opracować (wykłady : 4,5, instrukcja do lab1).

1. Jak jest przechowywany obiekt typu **Produkt1** w obiekcie typu **Fasada_warstwy_biznesowe**j?
2. Do czego służy metoda items() w klasie typu Fasada_warstwy_biznesowej? Należy okreslić, jakie dane są przetwarzane w tej metodzie oraz gdzie są wykorzystane dane utworzone przez metodę items()?
3. Jaki atrybut w klasie Managed_ produkt zawiera dane prezentowane za pomocą komponentu <h:DataTable> na stronie lista_produktow.xhtml?
4. Jak tworzona jest zawartość komponentu **<h:dataTable>** na stronie **lista_produktow.xhtml**? Należy podać, do czego służą atrybuty **items** oraz **item** znacznika <h:dataTable>:

   <h:dataTable value="#{**managed_produkt.items**}" **var="item".....**

   oraz jakie znaczniki są wykorzystane do tworzenia kolumnny w tabeli, jaki znacznik jest wykorzystany do tworzenia nagłówka tabeli oraz jaki znacznik służy do prezentacji danych w kolumnie tabeli. Nalezy to wykonać na podstawie znaczników podach poniżej:

```
<h:column>
        <f:facet name="header">
           <h:outputText value="#{bundle.Lista_produktow_nazwa}"/>
        </f:facet>
        <h:outputText value="#{item.get(0)}"/>
</h:column>
```

5. Kiedy na stronie **lista_produktow.xhtml** wyświetlany jest napis zdefiniowany w atrybucie **value** analizując fragment kodu strony JSF:

```
<h:outputText escape="false" value="#{bundle.Lista_produktow_pusta}"
              rendered="#{managed_produkt.items.rowCount == 0}"/>
```

6. Kiedy na stronie **lista_produktow.xhtml** wyświetlana zawartość tabeli analizując fragment strony lista_produktow.xhtml:

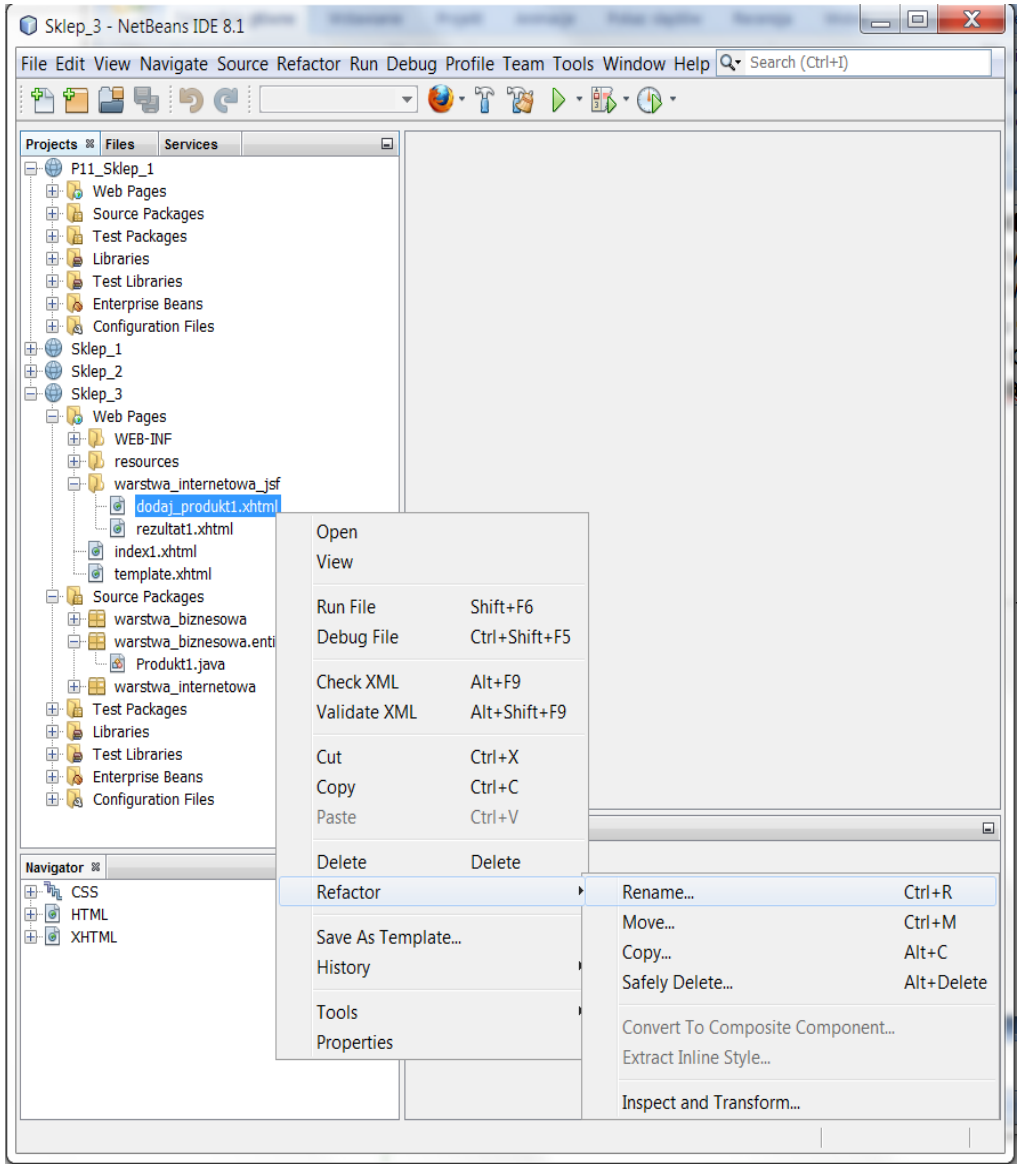
```
<h:panelGroup rendered="#{managed_produkt.items.rowCount > 0}">
   <h:dataTable value="#{managed_produkt.items}" var="item"......
```

7. Jaka rolę pełni plik **Bundle.properties** w budowie stron xhtml? Jak należy korzystać z tego pliku?

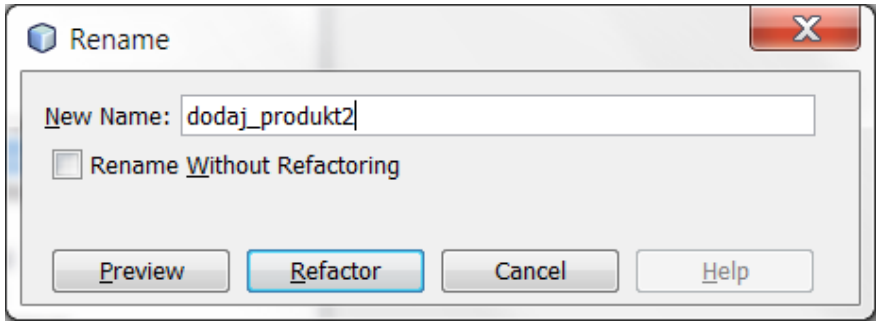
Przykład 3
Zastosowanie szablonu i znacznika **h:dataTable** do prezentowania zbioru produktów.

Należy w proponowanych zmianach w projekcie z przykładu 3 uwzględnić nowy/nowe atrybut/atrybuty obiektu typu Produkt1, wprowadzone podczas realizacji zadania w lab2.

1. **Tworzenie kopii projektu typu Web Application** o nazwie Sklep_3 z lab2 - Przykład1 (prawy klawisz myszy na nazwie projektu i wybór **Copy** – na formularzu kopiowania należy podać nową nazwę projektu **Sklep_3**. Projekt źródłowy **Sklep_2** należy zamknąć, spakować do formatu zip lub rar i usunąć wersję niespakowaną.



2. Należy zmienić nazwy plików **dodaj_produkt1** i **rezultat1** na **dodaj_produkt2** i **rezultat2** w folderze **warstwa_internetowa_jsf** następująco: kliknąć prawym klawiszem myszy na nazwę strony np. **dodaj_produkt1.xtml** i wybrać kolejno z list: **Refactor/Rename**. W formularzu **Rename**, w polu **New Name** należy wprowadzić nową nazwę: **dodaj_produkt2**. Podobnie należy zmienić nazwę strony **rezultat2.xhtml**.



2. Należy zmodyfikować kod klasy **Fasada_warstwy_biznesowej**

```
package warstwa_biznesowa;

import java.util.ArrayList;
import javax.ejb.Stateless;
import warstwa_biznesowa.entity.Produkt1;

@Stateless
public class Fasada_warstwy_biznesowej {

/*  pozostały kod bez zmian */

public ArrayList<ArrayList<String>> items() {
        ArrayList<ArrayList<String>> dane = new ArrayList();
         ArrayList<String> wiersz = new ArrayList();
        if (produkt!=null)
        {
          wiersz.add(produkt.getNazwa());
          wiersz.add("" + produkt.getCena());
          wiersz.add("" + produkt.getPromocja());
          wiersz.add("" + produkt.cena_brutto());
          dane.add(wiersz);
         }
        return dane;
    }
}
```

Dane przechowywanego obiektu typu **Produkt1** przeznaczone do prezentacji w komponencie dataTable – Jest to kolekcja elementów, które są kolekcją elementów typu String reprezentująca atrybuty i wyliczoną cenę brutto obiektu typu Produkt1

3. Należy zmodyfikować definicję klasy **Managed_produkt**

```
package warstwa_internetowa;

import warstwa_biznesowa.Fasada_warstwy_biznesowej;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;

@ManagedBean
@RequestScoped
public class Managed_produkt {

   @EJB
    private Fasada_warstwy_biznesowej fasada;
    private String nazwa;
    private String cena;
    private String promocja;
    private String cena_brutto;
    private DataModel items;

    public Managed_produkt() {    }

    public Fasada_warstwy_biznesowej getFasada()                 {  return fasada;    }

    public void setFasada(Fasada_warstwy_biznesowej fasada)   {  this.fasada = fasada;    }
```

DataModel – model danych komponentu dataTable

```
public String getNazwa() {
      return nazwa;
  }
public void setNazwa(String nazwa) {
      this.nazwa = nazwa;
  }
public String getCena() {
      return cena;
  }
public void setCena(String cena) {
      this.cena = cena;
  }
public String getPromocja() {
      return promocja;
  }
public void setPromocja(String promocja) {
      this.promocja = promocja;
  }
public String getCena_brutto() {
      return cena_brutto;
  }
public void setCena_brutto(String cena_brutto) {
      this.cena_brutto = cena_brutto;
  }
```

```
public DataModel utworz_DataModel() {
    return new ListDataModel(fasada.items());
}

public DataModel getItems() {
    if (items == null) {
        items = utworz_DataModel();
    }
    return items;
}

public void setItems(DataModel items) {
    this.items = items;
}
```

Utworzenie modelu komponentu **dataTable** na podstawie kolekcji zawierających elementy reprezentujące wiersz tabeli (kolekcja obiektów typu String reprezentująca atrybuty obiektu typu **Produkt** oraz cenę brutto)

Dodane metod do klasy **Managed_produkt** obsługujących dodawanie produktu (**dodaj_produkt**) po pobraniu danych z formularza za pomocą atrybutów: nazwa, cena, promocja i wywołaniu metody **utworz_produkt** ziarna EJB z obiektu fasada klasy typu Fasada_warstwy_biznesowej oraz wyświetlanie danych za pomocą metody **dane_produktu** pobranych z warstwy biznesowej od obiektu typu EJB fasada za pomocą metody **dane_produktu**

```
public String dodaj_produkt() {
        String[] dane = {nazwa, cena, promocja};
        fasada.utworz_produkt(dane);
        dane_produktu();
        return "rezultat2";
    }

public void dane_produktu() {
        String[] dane = fasada.dane_produktu();
        nazwa = dane[0];
        cena = dane[1];
        promocja = dane[2];
        cena_brutto = dane[3];
      }
}
```

4. Należy zmodyfikować definicję szablonu **template.xhtml** – dodanie w części przeznaczonej na menu (id=left) linku do strony **lista_produktow.xhtml** oraz znacznik **<h:panelGroup>** do obsługi wyświetlania błędów podczas wstawiania danych

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html">

  <h:head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
      <h:outputStylesheet name="css/default.css" />
      <h:outputStylesheet name="css/cssLayout.css"/>
      <title><ui:insert name="title">Facelets Template</ui:insert></title>
  </h:head>

  <h:body>
      <div id="top">
          <ui:insert name="top">Top</ui:insert>
      </div>
```

```
  <div>
    <div id="left">
     <h:link outcome="/faces/warstwa_internetowa_jsf/dodaj_produkt2"
                              value="Dodaj produkt"/><br/>
     <h:link outcome="/faces/warstwa_internetowa_jsf/lista_produktow"
                              value="Lista produktow"/>
    </div>
    <div id="content" class="left_content">
          <ui:insert name="content">Content</ui:insert>
     </div>
  </div>
  <h:panelGroup id="messagePanel" layout="block">
      <h:messages errorStyle="color: red" infoStyle="color: green" />
  </h:panelGroup>
  <div id="bottom">
        <ui:insert name="bottom">Bottom</ui:insert>
   </div>
 </h:body>
</html>
```

5. Należy w pliku kaskadowego arkusza stylu **cssLayout.css** w katalogu projektu: **Web Pages/resources/css (okno Projects)** wprowadzić zmianę stylu wyświetlania części strony:

```
<div id="content" class="left_content">
    <ui:insert name="content">Content</ui:insert>
</div>
```

w stylu o nazwie **.left_content**, aby ustalić obszar przeznaczony na część roboczą strony

```
.left_content {
    background-color: #dddddd;
    padding: 5px;
    margin-left: 170px;
}
```

```
.left_content {
    background-color: #dddddd;
    padding: 5px;
    margin-left: 170px;
    min-height: 50px;
}
```

6. Należy dodać do projektu stronę **lista_produktow.xhtml** opartą na szablonie **template.xhtml** i umieścić w katalogu **Web Pages/warstwa_internetowa_jsf** (okno **Projects)** – poniżej zawartość strony **lista_produktow.xhtml** do wyświetlania listy produktów dodana do projektu- **New/Other/JavaServer Faces/ Facelets Template Client...**

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core">

  <body>
    <ui:composition template="./../template.xhtml">
      <ui:define name="title">
            <h:outputText value="#{bundle.Lista_produktow_tytul}"></h:outputText>
      </ui:define>

      <ui:define name="content">
        <h:form>
      <h:outputText escape="false" value="#{bundle. Lista_produktow_pusta}"
                    rendered="#{managed_produkt.items.rowCount == 0}"/>
        <h:panelGroup rendered="#{managed_produkt.items.rowCount > 0}">
```

```
<h:dataTable value="#{managed_produkt.items}" var="item" border="0"
             cellpadding="2" cellspacing="0"
             rowClasses="jsfcrud_odd_row,jsfcrud_even_row"
             rules="all" style="border:solid 1px">

    <h:column>
            <f:facet name="header">
               <h:outputText value="#{bundle.Lista_produktow_nazwa}"/>
            </f:facet>
            <h:outputText value="#{item.get(0)}"/>
    </h:column>
    <h:column>
            <f:facet name="header">
               <h:outputText value="#{bundle.Lista_produktow_cena}"/>
            </f:facet>
            <h:outputText value="#{item.get(1)}"/>
    </h:column>
```

Komponent typu **dataTable** zbindowany z obiektem **items** typu **DataModel**

Item – element kolekcji items (zawierający dane atrybutów obiektu typu **Produkt oraz cenę brutto**

Nagłówek kolumny tabeli dataTable

Kolejny element kolekcji **item** (zawierający dane atrybutu **nazwa** obiektu typu **Produkt**), która jest elementem kolekcji **items** typu **DataModel**

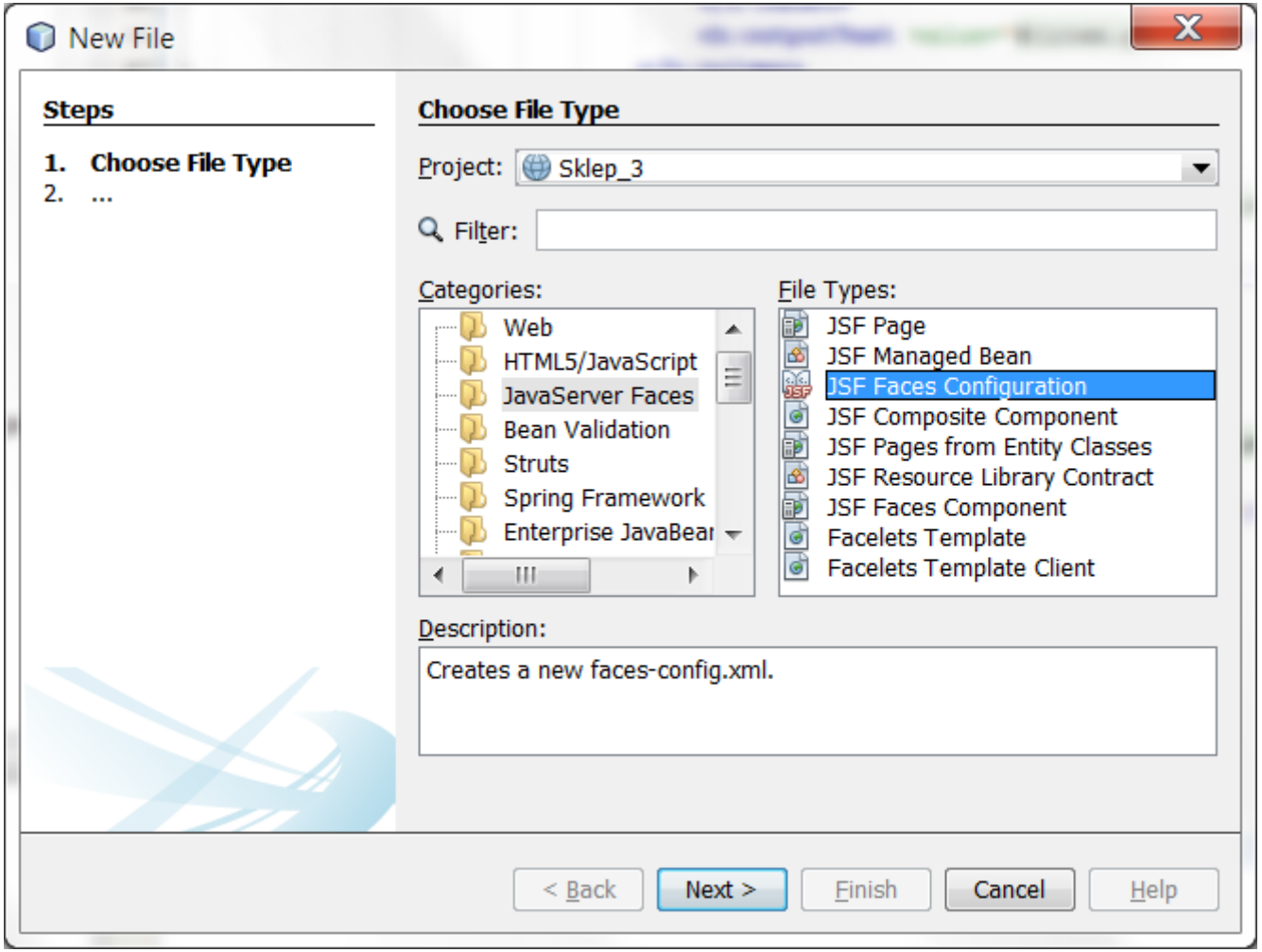
Kolejny element kolekcji **item** (zawierający dane atrybutu **cena** obiektu typu **Produkt**), która jest elementem kolekcji **items** typu **DataModel**

```
    <h:column>
        <f:facet name="header">
            <h:outputText value="#{bundle.Lista_produktow_promocja}"/>
        </f:facet>
        <h:outputText value="#{item.get(2)}"/>
    </h:column>

    <h:column>
        <f:facet name="header">
            <h:outputText value="#{bundle.Lista_produktow_cenabrutto}"/>
        </f:facet>
        <h:outputText value="#{item.get(3)}"/>
    </h:column>
  </h:dataTable>
 </h:panelGroup>
  <h:commandButton id="powrot" value="#{bundle.Lista_produktow_powrot}"
                   action="/faces/index1"/>
 </h:form>
</ui:define>

</ui:composition>

</body>
</html>
```

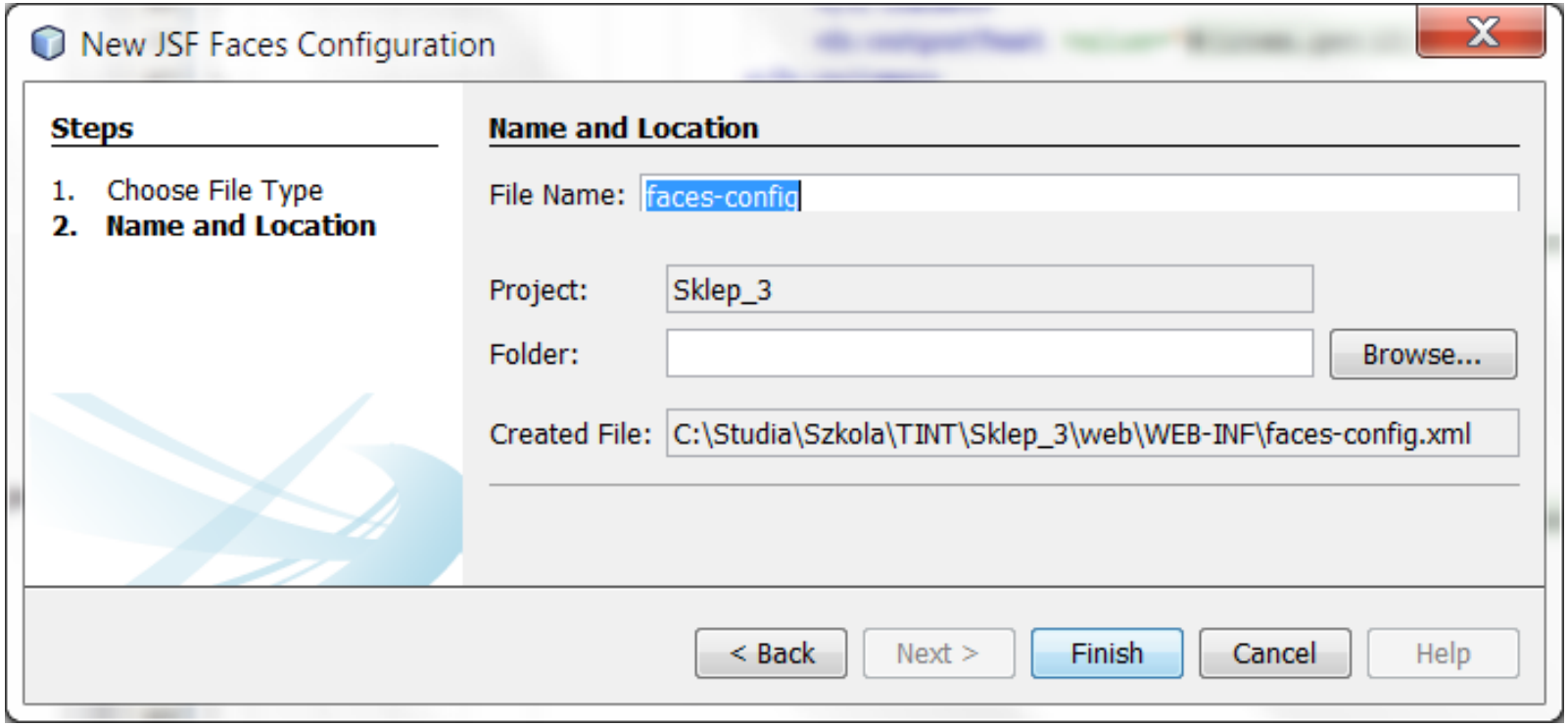
Kolejny element kolekcji **item** (zawierający dane atrybutu **promocja** obiektu typu **Produkt**), która jest elementem kolekcji **items** typu **DataModel**

Kolejny element kolekcji **item** (zawierający dane ceny brutto wyznaczanej przez metodę cena_brutto obiektu typu **Produkt**), która jest elementem kolekcji **items** typu **DataModel**

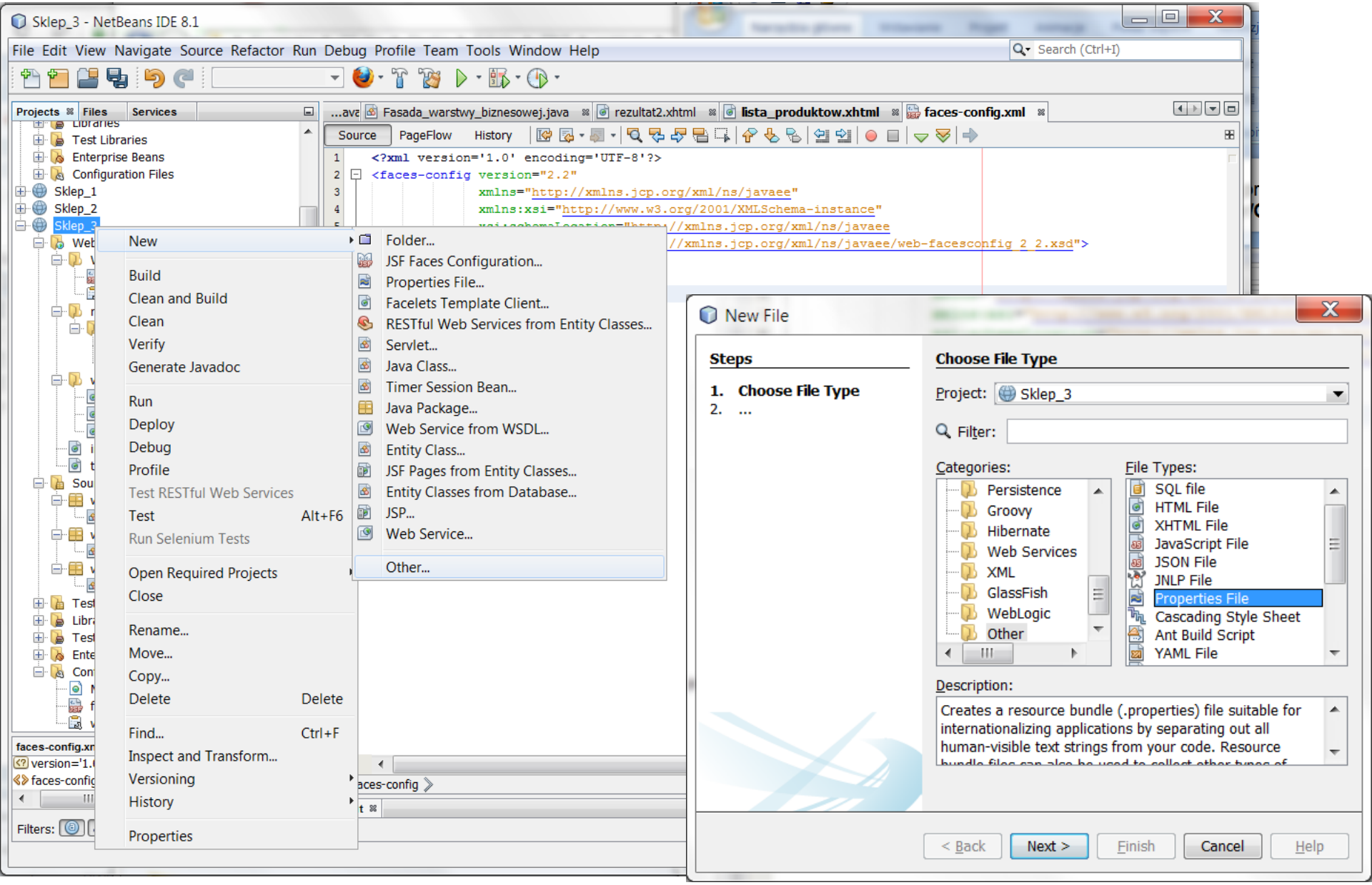
7. Dodanie pliku konfiguracji projektu **faces-config.xml** (**New/Other/JavaServer Faces/ JSF Faces Configuration**)



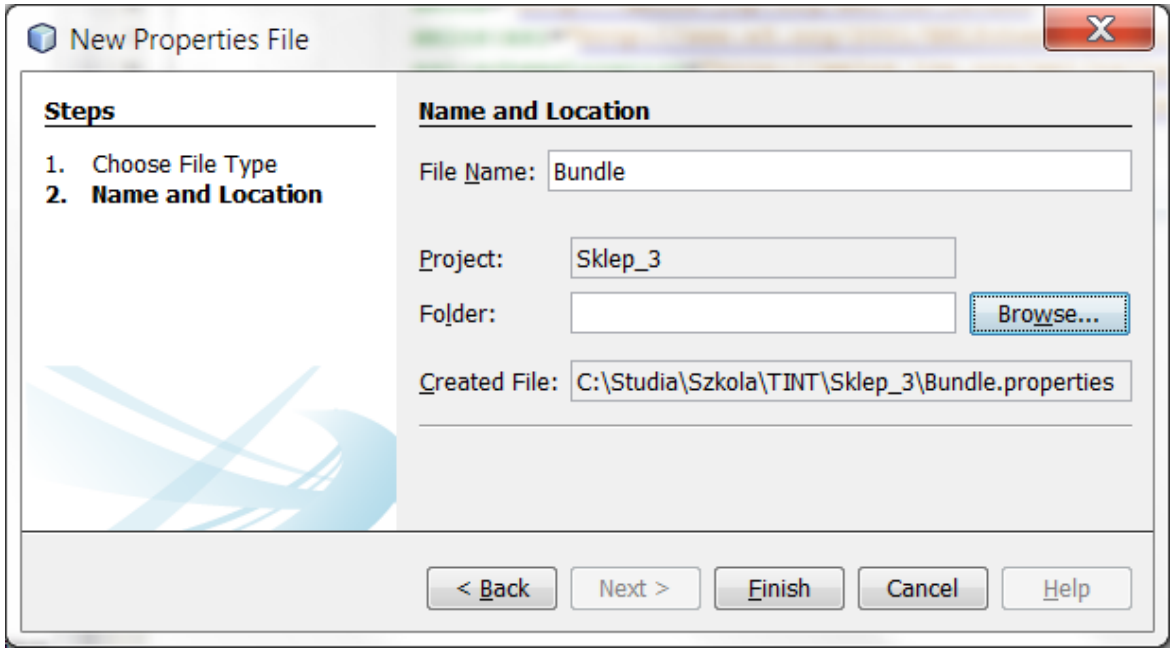
Dodanie pliku konfiguracji projektu **faces-config.xml**

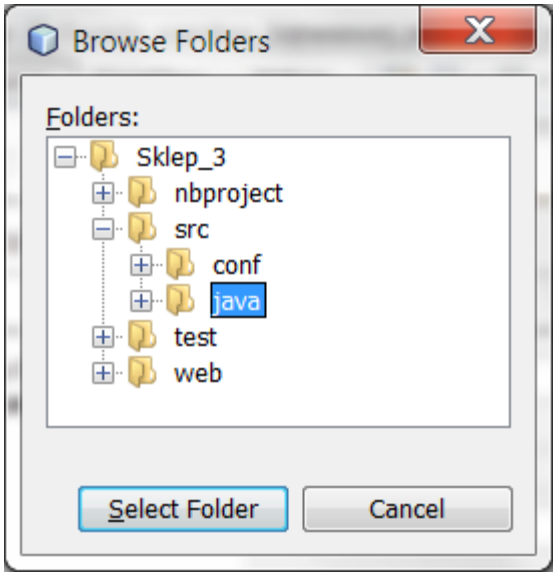


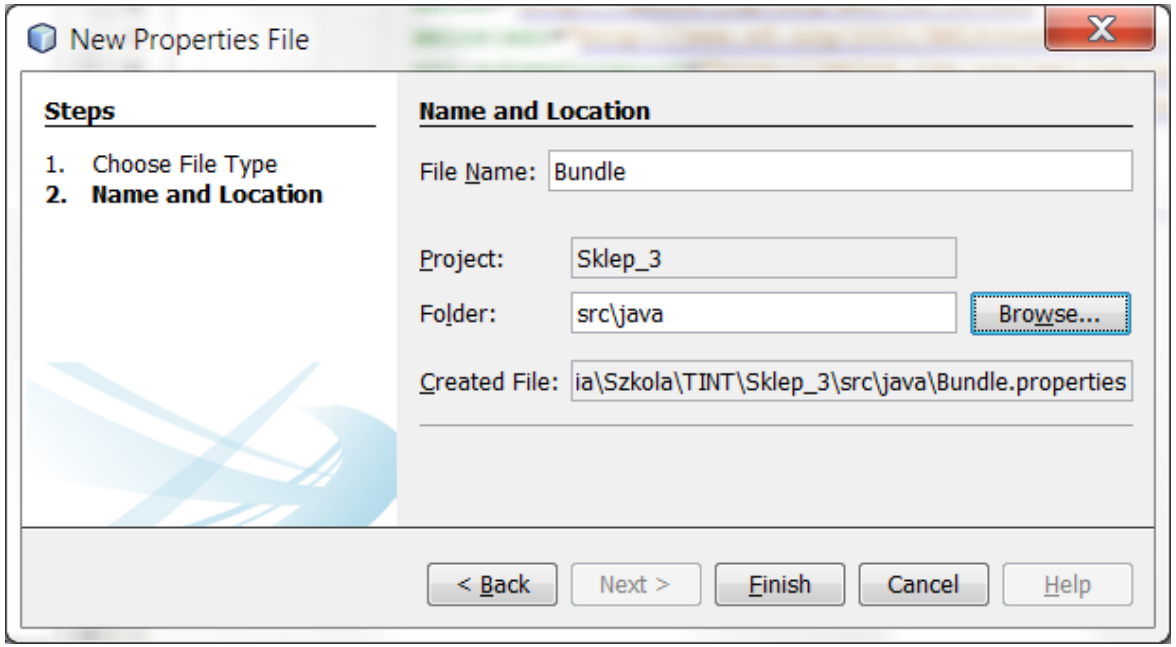
8. Dodanie pliku typu **properties** do projektu: prawy klawisz na **Source Packages, New/Other/Other/Properties File**



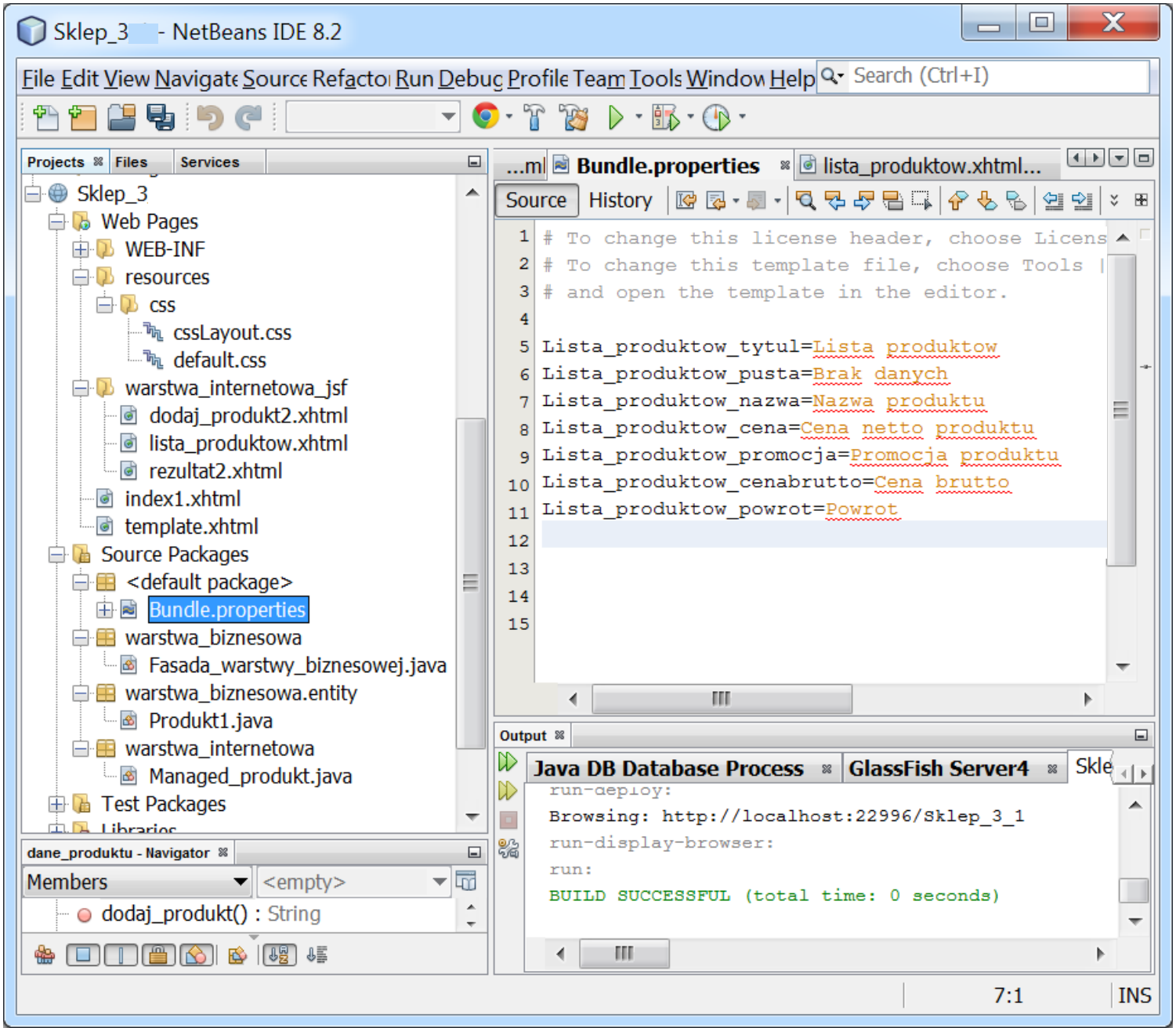
Dodanie pliku typu **properties** do projektu







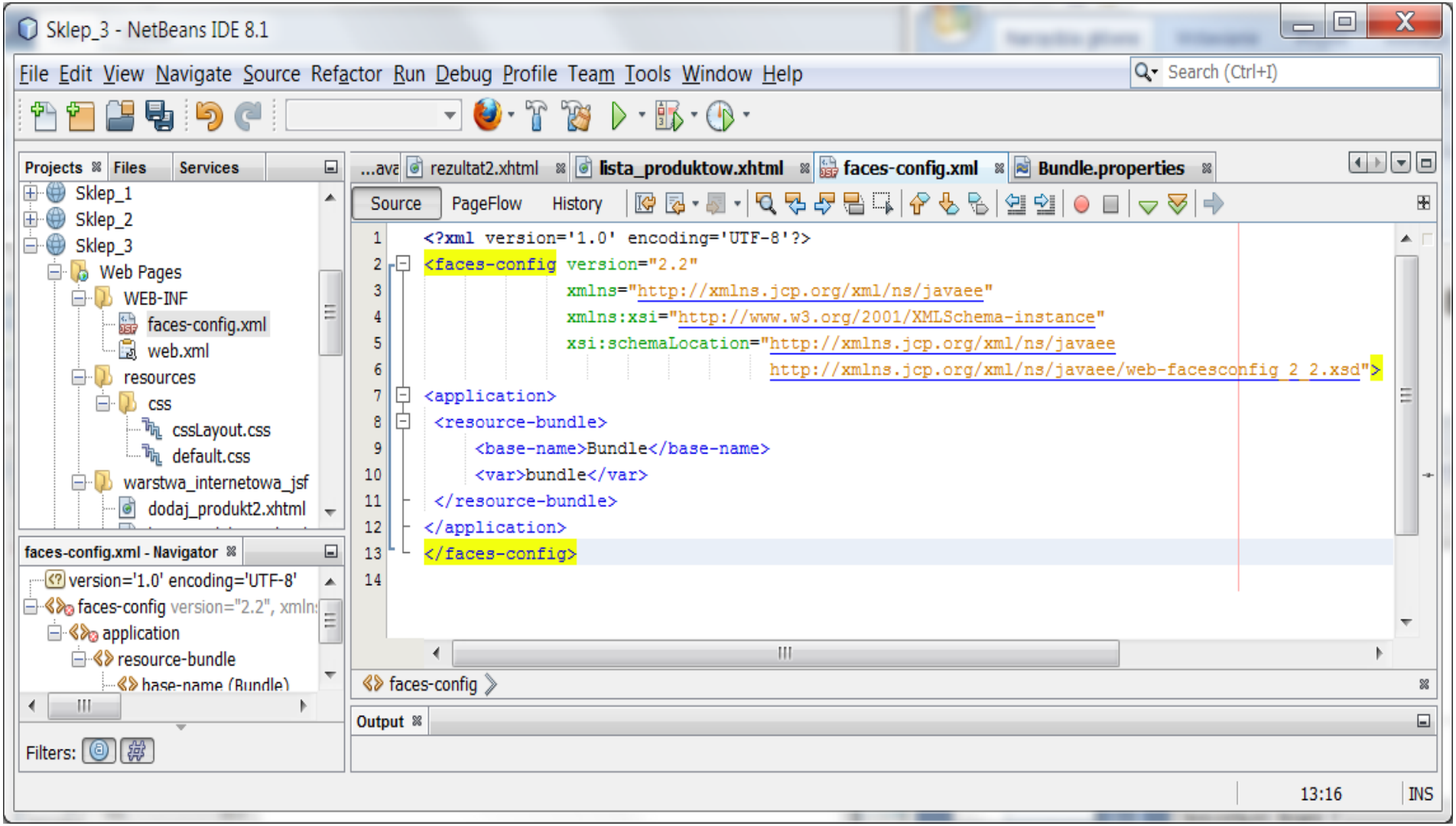
Zawartość pliku **Bundle.properties** zawierająca treść komunikatów.



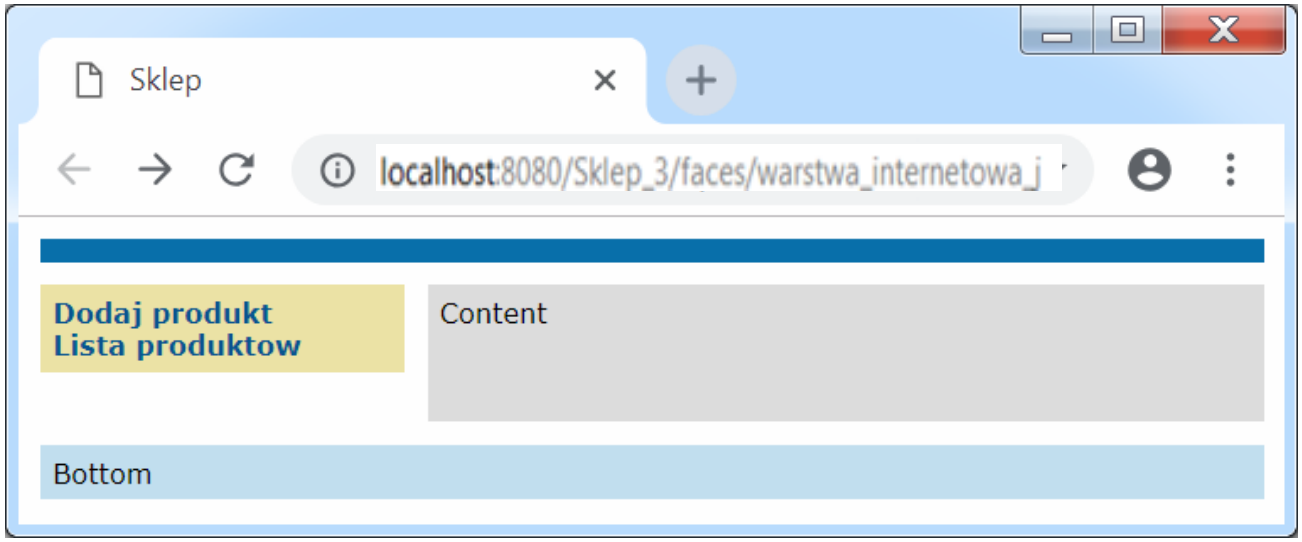
Zawartość pliku **Bundle.properties** zawierająca treść komunikatów używanych na stronie **lista_produktow.xhtml.** Należy w taki sam sposób zastąpić komunikaty w pozostałych plikach xhtml

```
Lista_produktow_tytul=Lista produktow
Lista_produktow_pusta=Brak danych
Lista_produktow_nazwa=Nazwa produktu
Lista_produktow_cena=Cena netto produktu
Lista_produktow_promocja=Promocja produktu
Lista_produktow_cenabrutto=Cena brutto
Lista_produktow_powrot=Powrot
```

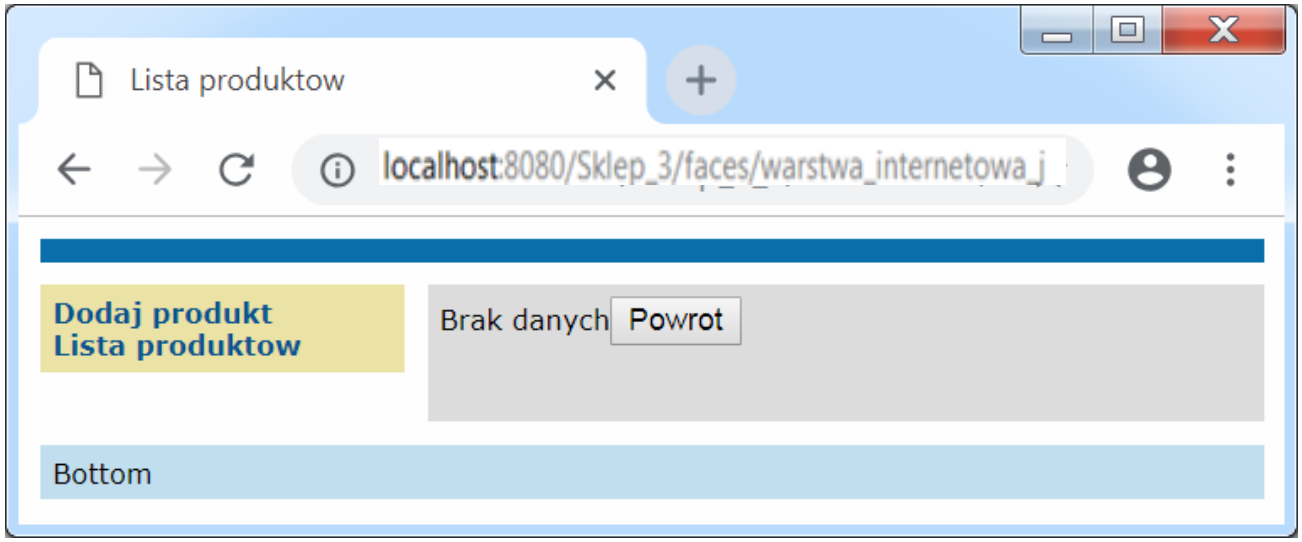
9. Należy zadeklarować plik **Bundle.properties** w pliku konfiguracyjnym **faces-config.xml.**



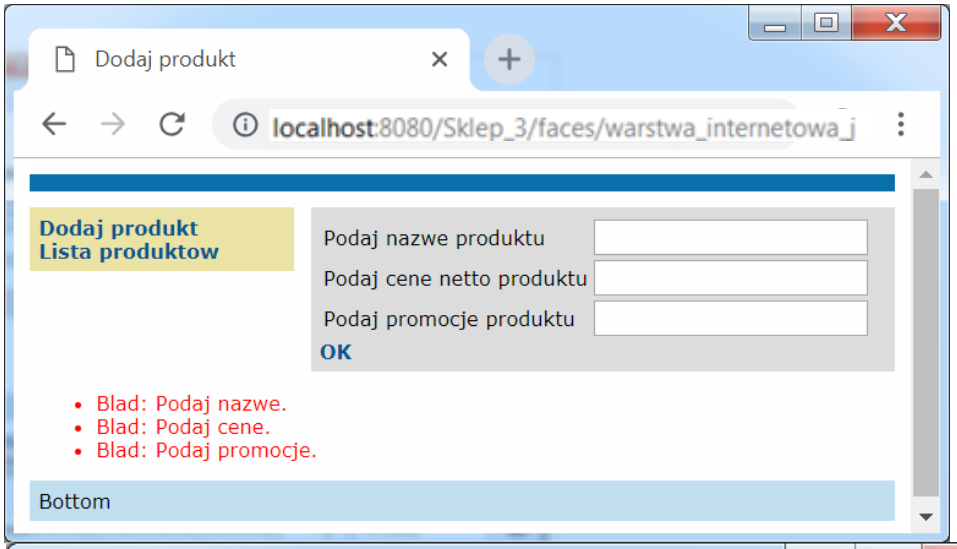
10. Uruchomienie aplikacji



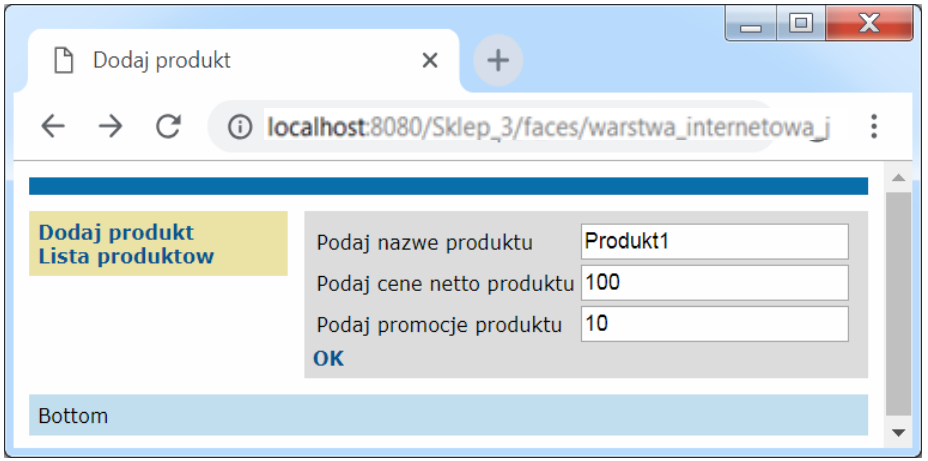
Widok strony lista_produktow.xhtml po kliknięciu na **Lista produktow**

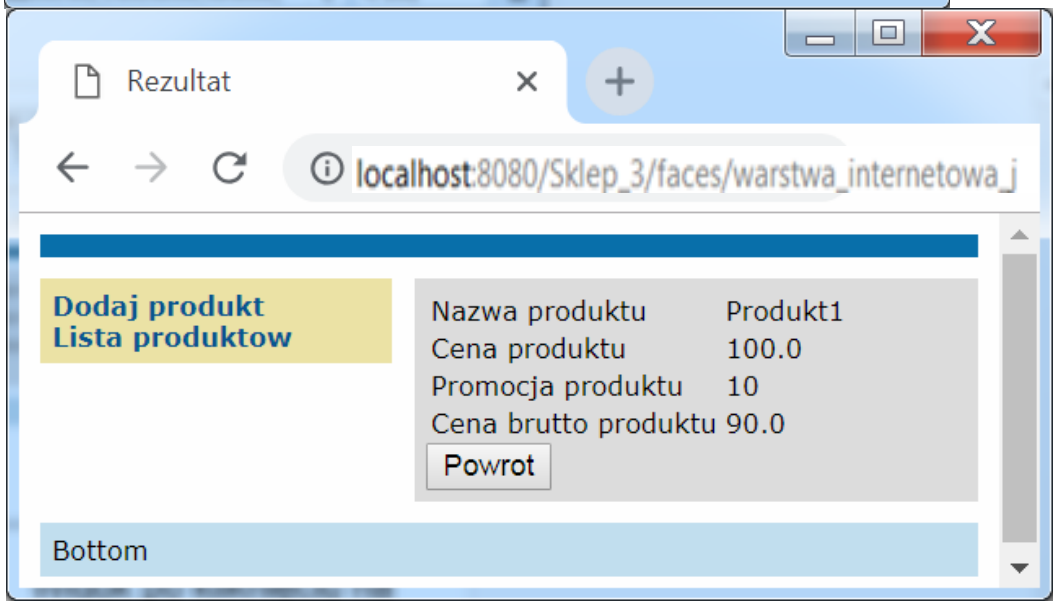


Widok po kliknięciu na **Dodaj produkt** na stronie **index1.xhtml** i kliknięciu na przycisk **OK** bez wprowadzenia danych produktu

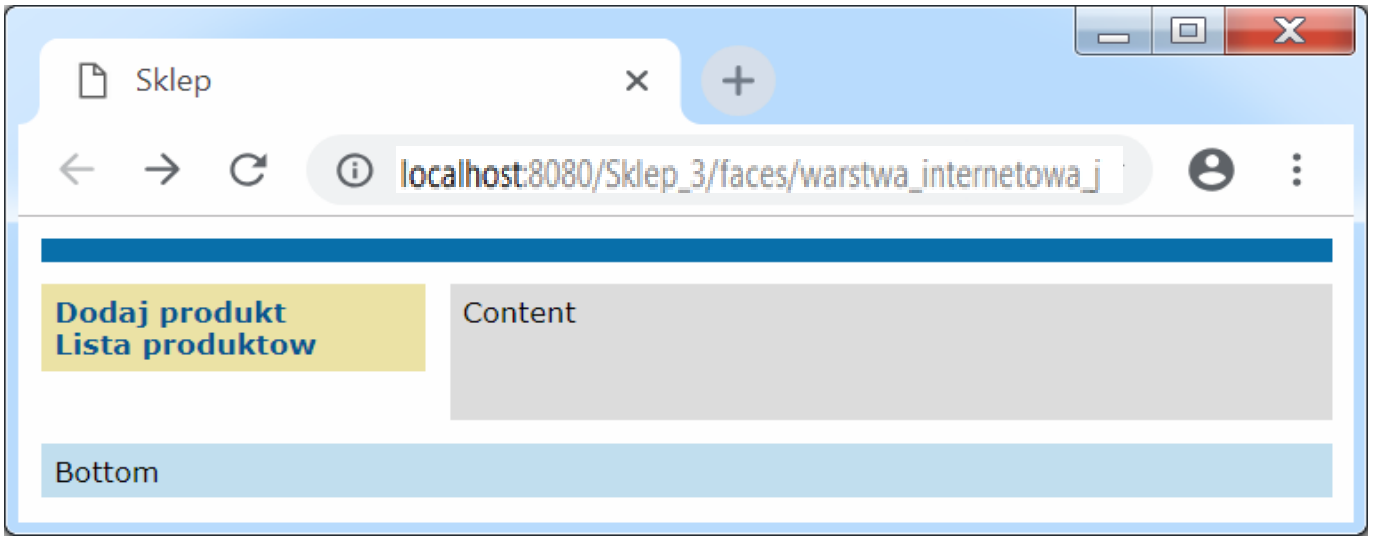


Widok po kliknięciu na **Dodaj produkt** na stronie **index1.xhtml** i wprowadzeniu danych

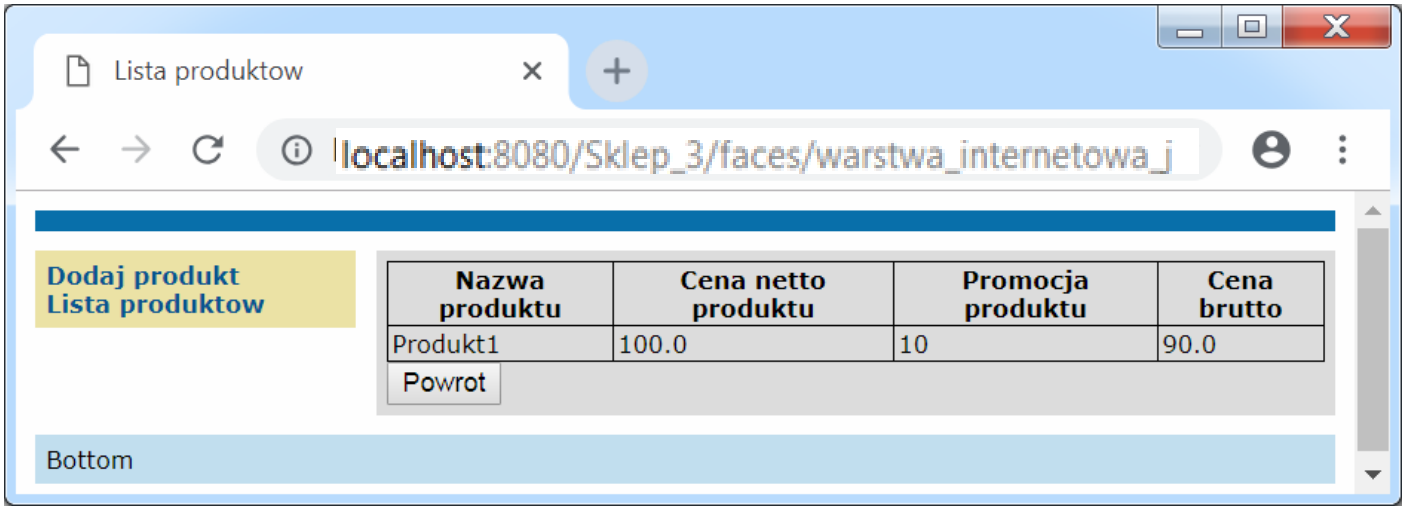




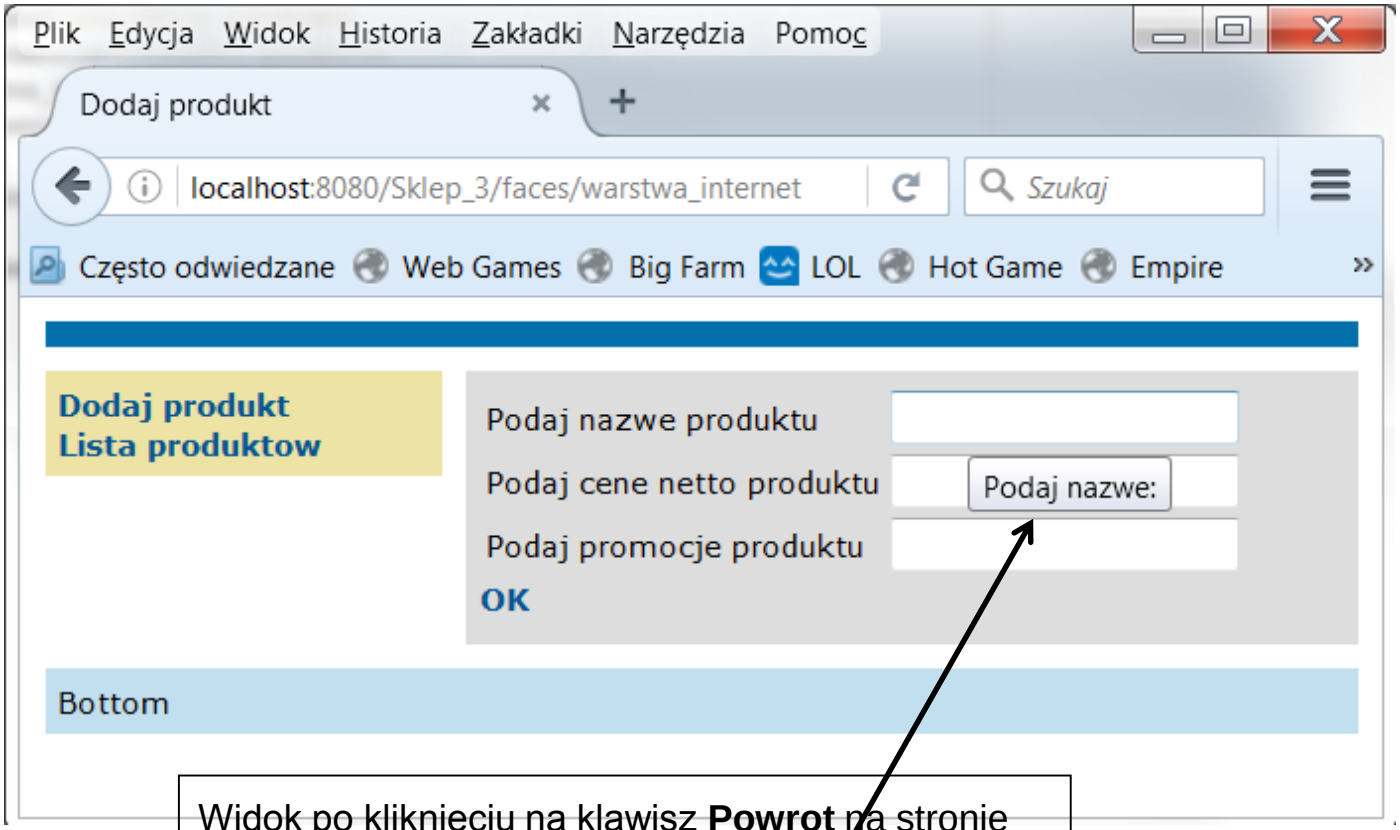
Widok po kliknięciu na **Ok** na stronie **dodaj_produkt2.xhtml** po wprowadzeniyudanych produktu



Widok po kliknięciu na **Powrot na** stronie **rezultat2.xhtml**



Widok po kliknięciu na **Lista produktow** na stronie **index1.xhtml**



Widok po kliknięciu na klawisz **Powrot** na stronie **lista_produktow.xhtml**, następnie wybranie pozycji z lewej strony okna: **Dodaj produkt i** po położeniu kursora myszy na polu z etykietą **Podaj nazwe produktu –** wyświetlenie zawartości atrybutu **tytul** znacznika **<h:inputTytul>**

```
<h:inputText
    id="nazwa"
    title="Podaj nazwe:"
    value="#{managed_produkt.nazwa}"
    required="true"
    requiredMessage="Blad: Podaj nazwe." >
</h:inputText>
```

11. Należy przenieść wszystkie komunikaty do pliku **Bundle.properties** ze stron **dodaj_produkt2.xhtml**, **rezultat2.xhtml,** rozróżniając w nazwie komunikatu przynależność do strony. Część komunikatów strony **lista_produktow.xhtml** można wykorzystać na stronie **rezultat2.xhtml.** Poniżej pokazano fragment pliku typu **properties** oraz przykład zastosowania komunikatów z tego pliku. Następny slajd pokazuje kolejny przykład wykorzystania pliku typu **properties**.

Fragment zawartości pliku **Bundle.properties** po proponowanych zmianach

```
lista_produktow.tytul=Lista produktow
lista_produktow.pusta=Brak danych
lista_produktow.nazwa=Nazwa produktu
lista_produktow.cena=Cena netto produktu
lista_produktow.promocja=Promocja produktu
lista_produktow.cena_brutto=Cena brutto
lista_produktow.powrot=Powrot
```

Fragment zawartości pliku **lista_produktow.xhtml** po proponowanych zmianach – zmiana odwołania do komunikatu w postaci indeksowania nazwą komunikatu zawartości pliku **Bundle.properties**. Podobne odwołania należy wykonać we wszystkich plikach xhtml, zawierających komunikaty (atrybuty: **value**, **requiredMessage**)

```
<body>
  <ui:composition template="./../template.xhtml">
     <ui:define name="title">
        <h:outputText  value="#{bundle['lista_produktow.tytul']}" />
     </ui:define>
     <ui:define name="content">
        <h:form styleClass="jsfcrud_list_form">
           <h:outputText escape="false"  value="#{bundle['lista_produktow.pusta']}"
                         rendered="#{managed_produkt.items.rowCount == 0}"/>
```