

Wykład Java Server Pages (JSP)

Zaawansowane programowanie w Javie

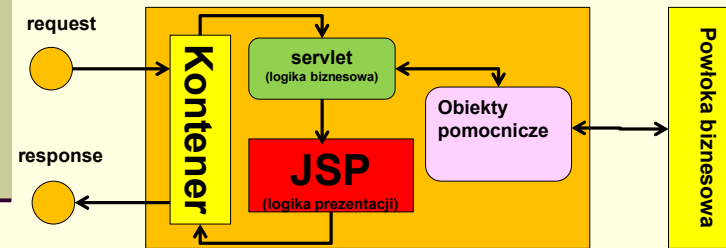
Dr inż. Damian Raczyński

Wprowadzenie – czym jest JSP



Widok dynamicznych elementów

Obsługuje logikę prezentacji w modelach MVC



3

Czego można się spodziewać na wykładzie

1. Wprowadzenie,
2. Elementy JSP,
3. Obiekty bezpośrednie (Implicit Objects),
4. Biblioteki znaczników,
5. Akcje JSP,
6. Obsługa błędów.

2

Do czego można używać serwletów?



- uwierzytelnianie użytkowników,
- odwołania do bazy danych,
- filtrowanie przychodzących żądań od użytkowników,
- logika w aplikacji,
- przekierowanie do JSP.

4

Wady servletów

Tworząc odpowiedź (stronę Internetową) z wykorzystaniem servletów konieczne jest umieszczanie kodu tworzonej strony w instrukcjach języka Java – przykładowo:

```
out.print("<h1> nagłówek </h1> ...");
```

Tworzony servlet stanowi zatem kod Javy z „wklejonymi” znacznikami HTML.

Odwrotność do servletów stanowi technologia JSP – wykorzystywany jest HTML z „wklejonym” kodem Java

5

JSP

Servlety

Obie technol
loginy użytko
Przeznaczeni

Metody cyklu życia:
init() – może zostać nadpisana
service() – może zostać nadpisana
destroy() – może zostać nadpisana

język HTML wewnątrz JAVY

Generalnie można przyjąć zasadę, że metody, których nazwa rozpoczyna się od podkreślenia nie mogą zostać nadpisane.

Metody cyklu życia:
jspinit() – może zostać nadpisana
_jspService() – nie może zostać nadpisana
jspDestroy() – może zostać nadpisana

JAVA wewnątrz HTML

Obie technologie działają wyłącznie w obrębie kontenera Web

JSP

Java Server Pages (JSP) jest wykorzystywane do prezentacji informacji. JSP stanowi także servlet.

JSP umożliwia zamieszczanie zwyczajnego kodu HTML oraz informacji generowanych przez serwlety.

Dokumenty JSP są automatycznie przekształcane do postaci serwletów.

Oprócz kodu HTML w JSP mogą pojawić się:
elementy skryptowe,
dyrektywy,
akcje.

6

JSP

JSP tak na prawdę jest zwykłym servletem!

Transformacja

Tłumaczenie

Po załadowaniu przez web kontener

JSP
np.
str.jsp

servlet
str_jsp.java

kod bajtowy
00101010100
0110
str_jsp.class

servlet
str_jsp
Servlet

8

JSP



Jakie elementy strony JSP zapewniają dynamiczną zawartość ????

Deklaracje JSP – kod, który trafia na zewnątrz metody `service`

Skryptlety JSP - elementy stanowią kod Javy realizujący pewną funkcjonalność w metodzie `service`.

Wyrażenia JSP – kod wyrażen jest wyznaczany

Dyrektwy JSP - pozwalają na określenie struktury generowanego serwletu.

Akcje JSP - rozszerzenie funkcjonalności.

9

Elementy skryptowe



Deklaracje:

`<%! KOD_JAVA %>` - znacznik deklaracji

Wszystko, co zostanie zamieszczone w znaczniku zostaje umieszczone na zewnątrz metody `service`

Co tu zamieścić?

Deklarację metod i zmiennych

Po co używać znacznika?

Przykłady – zmienne współdzielone pomiędzy żądaniami, powtarzający się kod w JSP można zamieścić w metodzie

11

Elementy skryptowe



Jakie elementy strony JSP zapewniają dynamiczną zawartość ??? - cd

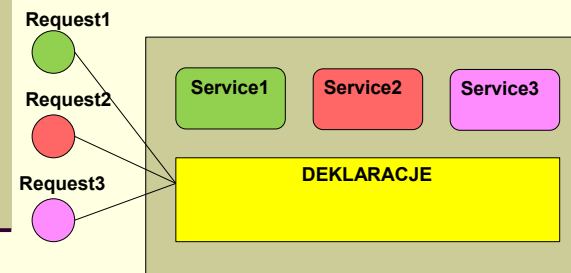
Wyrażenia: `<%= wyrażenie %>` - wartość jest obliczana i wstawiana do kodu

skryptlety: `<% kod %>` - kod wstawiany do metody `_jspService` serwletu

deklaracja: `<%! kod %>` - kod wstawiany wewnątrz klasy serwletu (poza metodami).

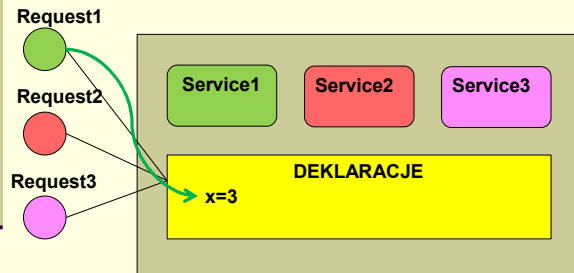
10

Elementy skryptowe



12

Elementy skryptowe



13

Elementy skryptowe



Skryptety:

Znacznik `<% KOD_JAVY %>` nazywany jest skryptetem.

Wszystko co zostanie zawarte w tym znaczniku wędruje do metody `service`

Wszystkie zmienne zadeklarowane w skrypcie są zmiennymi lokalnymi metody!

Co należy umieścić w skrypcie?

Logikę biznesową po stronie JSP

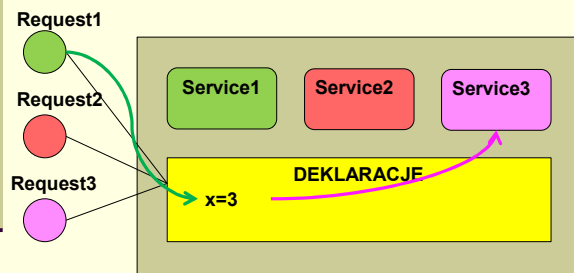
Po co skryptety są potrzebne?

Wykonanie niewielkich operacji związanych z logiką biznesową (jeżeli operacje są skomplikowane powinny zostać umieszczone w serwlecie)

Wykonanie prostej walidacji danych

15

Elementy skryptowe



Zmienne zadeklarowane w blokach deklaracji są widoczne pomiędzy zadaniami!!!

14

Elementy skryptowe



Wyrażenia:

Znaczniki `<%= KOD_JAVY%>` nazywane są znacznikami wyrażen

Kod javy umieszczony w wyrażeniu jest wykonywany, natomiast zwrócone wartości wyświetlane są na stronie

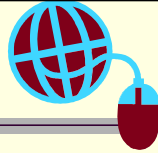
Cokolwiek jest umieszczane wewnątrz wyrażenia powinno zwracać wartość.

Co umieszczać wewnątrz wyrażen?

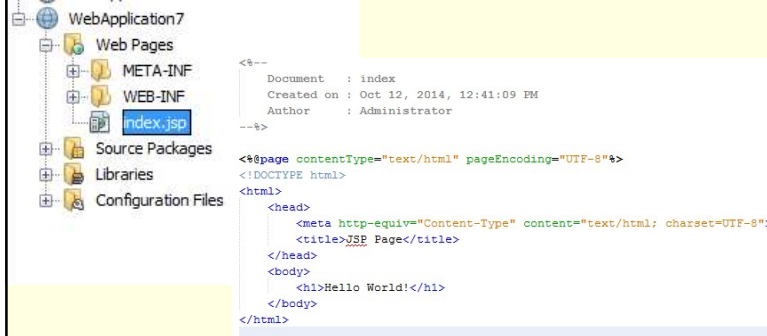
Zmienne lub metody zwracające wartość

16

JSP



Tworzenie strony JSP:



17

Elementy skryptowe



Predefiniowane zmienne:

Zmienna	Znaczenie
request	obiekt <code>HttpServletRequest</code>
response	obiekt <code>HttpServletResponse</code>
session	obiekt <code>HttpSession</code>
out	obiekt <code>JspWriter</code>

19

Elementy skryptowe



Przykład:

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<h1>Hello World!</h1>
<p>
<%= new java.util.Date() %>
</p>
</body>
</html>

```

18

Elementy skryptowe



Przykład:

```

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<h1>Hello World!</h1>
<p>
<%= request.getRemoteHost() %>
</p>
</body>
</html>

```

20

Elementy skryptowe



Skryptlety pozwalają na wstawienie dowolnego kodu Java:
`<% kod java %>`.

Przykład: np. w jaki sposób ustawić kolor tła strony Internetowej na wartość przekazaną jako parametr URL?

Przykład w NetBeans

21

Elementy skryptowe



Skryptlety mogą zostać również wykorzystane do warunkowego przetwarzania strony Internetowej.

Konstrukcja typu:

```
<% if (warunek) { %>
    HTML
<% } else { %>
    HTML
<% } %>
```

Wykona wyłącznie jeden z bloków HTML.

23

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <%
    String kolor=request.getParameter("kolorTla");
    boolean kolorPodany;
    if(kolor!=null) kolorPodany=true;
    else { kolorPodany=false; kolor="WHITE"; }
  %>

  <body bgcolor="<%= kolor %>">
    <h1>Hello World!</h1>
    <p>
      <%= request.getParameter("kolorTla") %>
    </p>
  </body>
</html>
```

Hello World!

```
<body>
  <h1>Rzut kostką gra v 1.0</h1>
  <%
    int losowa = (int) Math.floor(Math.random()*10000)%6+1;
    String strzal=request.getParameter("strzal");
    int podana=Integer.parseInt(strzal);
  %>
  <p>
    Podałeś: <%= podana %>
    Wylosowano: <%= losowa %>
    <%if(podana==losowa){ %>
      <b> BRAWO !!!!!!!!!!!!!!! </b>
    <% }else{ %>
      <b> nie zniechęcaj się ! </b>
    <% } %>
  </p>
</body>
```

Deklaracje



Deklaracje JSP – mechanizm deklarowania metod i atrybutów umieszczonych bezpośrednio w klasie servletu (NIE W METODZIE `_jspService`).

Zadeklarowane atrybuty „SĄ WIDOCZNE” RÓWNIEŻ PO OBSŁUŻENIU DANEGO ŻĄDANIA.

```
<%! operator_widoczności typ dana = wartość; %>
```

- kod wykonywany tylko jeden raz! – należy traktować jako deklarację atrybutu w klasie servletu (deklaracja nie powtórzy się przy wywołaniu kolejnego żądania, dopiero po ponownym uruchomieniu serwera)

25

Deklaracje



```
<body>
  <h1> Prosty licznik odwiedzin (a raczej wywołań strony JSP) </h1>
  <%! private int licznikOdwiedzin=0; %>
  <p>
    Od ostatniego uruchomienia serwera
    strona została wywołana: <%= ++licznikOdwiedzin %> razy
  </p>
</body>
```

27

Deklaracje



Przykład w NetBeans – licznik odwiedzin (do uruchomienia serwera)

26

Deklaracje



Podobnie wygląda deklaracja metod:

```
<%!
    operator_wid. typ nazwa(parametry)
    {
        Kod funkcji
    }
%>
```

28

Deklaracje



```
<body>
<h1> Liczenie wartości n'tej liczby Fibonacciego </h1>
<%!
    private int fibonacci(int n)
    {
        if (n==1)
            return 0;
        else if (n==2)
            return 1;
        else return fibonacci(n-1)+fibonacci(n-2);
    }
%>
<p>
    <% String parametr=request.getParameter("fibon");
    int liczba=Integer.parseInt(parametr);
    %>
    Liczba Fibonacciego wynosi: <%= fibonacci(liczba) %>
</p>
</body>
```

Dyrektywy



W JSP można stosować 3 dyrektywy:

page,
include,
taglib.

page – umożliwia importowanie klas, modyfikację klasy bazowej serwletu, określanie typu zawartości

include – umożliwia wstawianie zawartości plików do klasy serwletu.

taglib – służy do definiowania własnych znaczników.

31

Dyrektywy



Dyrektywa to nazwa różnych konstrukcji, które łączy składnia – sposób zapisu w JSP.

Dostarczają one informacji wykorzystywanych w procesie preprocesingu stron JSP (dyrektywy są przetwarzane przed przetworzeniem strony JSP na serwlet)

Określają one informacje o stronie, które pozostają niezmienione między żądaniami.

<%@ kod dyrektywy %>

- znaczniki tego typu mogą zostać umieszczane w każdym miejscu strony JSP.

30

Dyrektywy



<%@ page import	= „{importowane klasy}”
contentType	= „{Content type}”
isThreadSafe	= „{true/false}”
session	= „{true/false}”
buffer	= „{rozmiar bufora}” <small>SBK default.</small>
extends	= „{dziedziczenie_po_stronie}”
info	= „{informacje o stronie}”
errorPage	= „{nazwa strony błędu}”
isErrorPage	= „{true/false}”
language	= „{język}” <small>java default.</small>

32

Dyrektywy



import	importowanie np. klas bibliotek
contentType	html, zip, film, wideo, ...
isThreadSafe	jeżeli false – kontener nie będzie wywoływał serwletu w wątkach (obsługa żądań jednowątkowa z oczekiwaniem)
session	czy można używać bieżącej sesji HTTP czy nie
errorPage	Po wystąpieniu błędu użytkownik Zostanie przekierowany na konkretną stronę
isErrorPage	Czy strona błędu
language	Język – domyślnie java

33

Dyrektywy



`<%@ page import ... %>, np:`
`<%@ page import="java.util.*,coreservlets.*" %>`

Służy do importowania używanych bibliotek. Przykładowo:

```
<body>
  <%@ page import="java.util.*" %>
  <p>
    <%
      Random a = new Random();
      int u= a.nextInt() %5;
    %>
    Wartość losowa wynosi: <%= u %>
  </p>
</body>
```

35

Dyrektywy



Pozostałe dyrektywy:

- **extends** – klasa bazowa serwletu (ostrożnie),
- **info** – łańcuch znaków, jaki zostanie wygenerowany w wyniku wywołania metody `getServletInfo`,
- **errorPage** – strona JSP, która ma zostać użyta do obsługi zgłoszonych wyjątków,
- **isErrorPage** – określa, czy strona jest używana przez inne do obsługi błędów.

34


Dyrektywy



```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1> tytuł strony </h1>

  </body>
</html>
```


36



```
<%@page contentType="application/vnd.ms-excel" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>

  <body>
    <TABLE BORDER=1>
      <TR><TH></TH><TH>Jabka<TH>Pomarańcze
      <TR><TH>Pierwszy kwartał<TD>2307<TD>4706
      <TR><TH>Drugi kwartał<TD>2982<TD>5104
      <TR><TH>Trzeci kwartał<TD>3011<TD>5220
      <TR><TH>Czwarty kwartał<TD>3055<TD>5287
    </TABLE>

  </body>
</html>
```



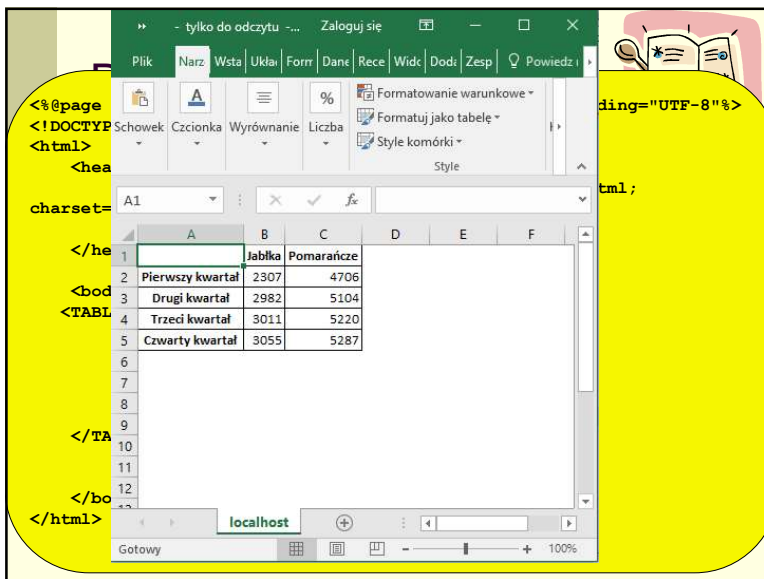
Dyrektywy

Dyrektywa include określa dołączane pliki do głównej strony JSP.

Składnia:
`<%@ include file=„względny_adres” %>`

Jeżeli dołączany plik zostanie zmieniony, to trzeba będzie zmodyfikować wszystkie strony JSP.


39



```
<%@page contentType="application/vnd.ms-excel" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>

  <body>
    <TABLE BORDER=1>
      <TR><TH></TH><TH>Jabka<TH>Pomarańcze
      <TR><TH>Pierwszy kwartał<TD>2307<TD>4706
      <TR><TH>Drugi kwartał<TD>2982<TD>5104
      <TR><TH>Trzeci kwartał<TD>3011<TD>5220
      <TR><TH>Czwarty kwartał<TD>3055<TD>5287
    </TABLE>

  </body>
</html>
```



Dyrektywy

Dyrektywa include dołącza pliku w chwili tłumaczenia

Zawartość główna

```
<%@ include file=„naglowek.jsp"%>
Witaj <%=userName%>

<%@ include file=„stopka.jsp"%>
<table><tr>
  <td> KONTAKT </td>
</tr></table>
```

naglowek.jsp

```
Witaj <%=userName%>
```

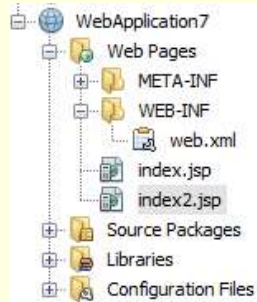
stopka.jsp

```
<table><tr>
  <td> KONTAKT </td>
</tr></table>
```

40

Dyrektywy

Przykładowe łączenie dwóch stron:



41

Dyrektywy

```

index2.jsp:
<%@page import="java.util.Date"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <hr/>
    <hr/>
    <h1> strona numer 2 </h1>
    <p> ta strona jest dołączona do strony głównej </p>
    <h1> Prosty licznik odwiedzin (a raczej wywołan strony JSP) </h1>
    <%! private int licznikOdwiedzin=0; %>
    <p>
      Od ostatniego uruchomienia serwera
      strona została wywołana: <%= ++licznikOdwiedzin %> razy
    </p>
  </body>
</html>
  
```

Dyrektywy

```

index.jsp:
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;
    charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Strona główna !!!!!!! </h1>
    <%@ include file="index2.jsp" %>
  </body>
</html>
  
```

Komunikacja serwletów ze stroną JSP

Komunikacja serwletów ze stronami JSP odbywa się za pośrednictwem atrybutów.

Przykładowo:

```

request.setAttribute("nazwa", wartość);

request.getAttribute("nazwa");
  
```

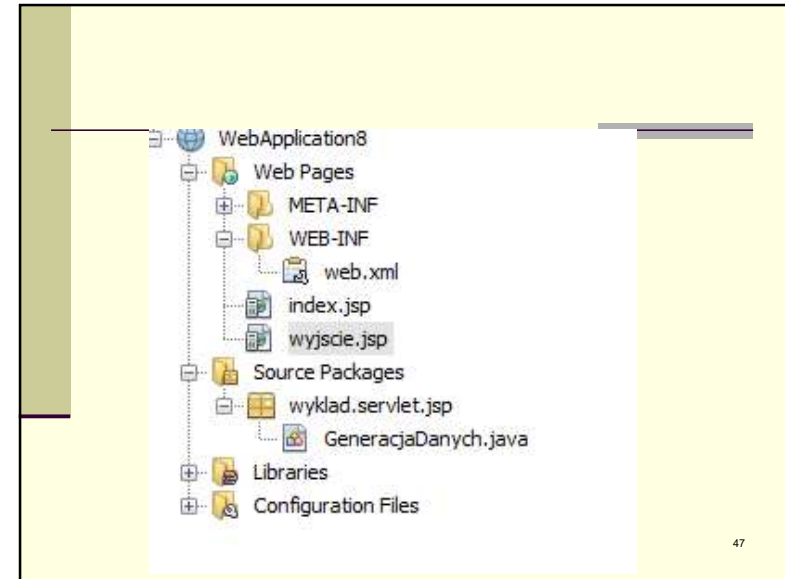
44

Komunikacja serwletów ze stroną JSP



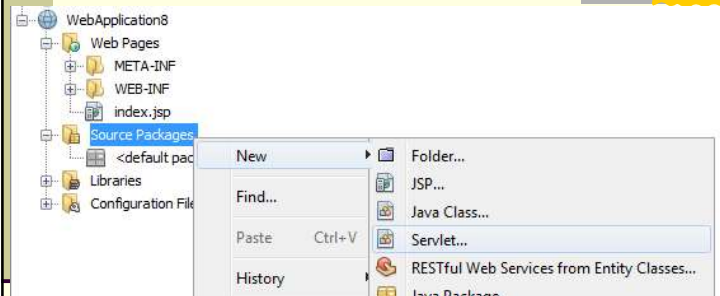
Przykład w NetBeans – komunikacja serwlet→JSP

45



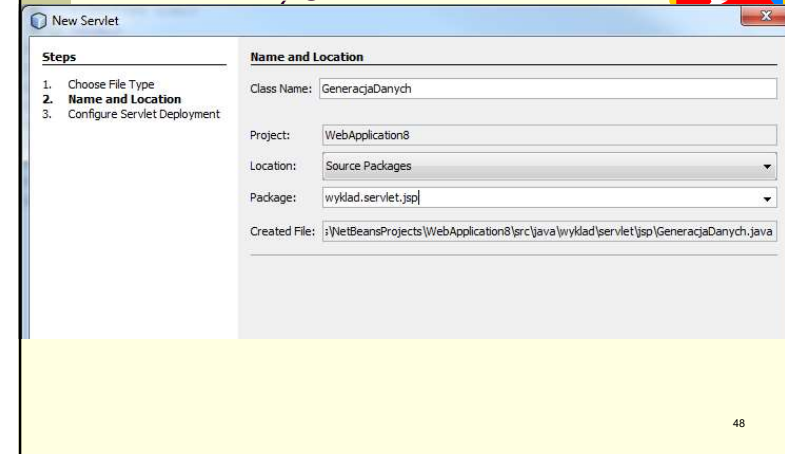
47

Komunikacja serwletów ze stroną JSP

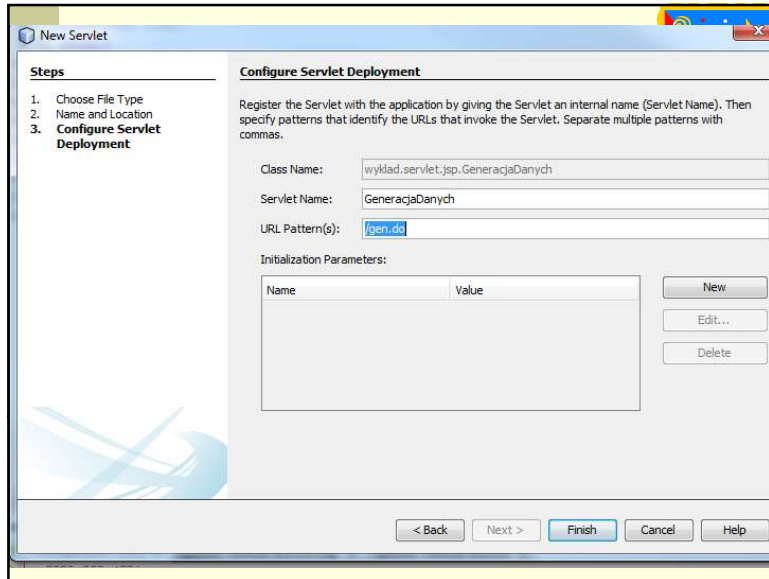


46

Komunikacja serwletów ze stroną JSP



48



Komunikacja serwletów

```

servlet GeneracjaDanych.java:
protected void processRequest(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        String param=request.getParameter("pole1");
        int ile=Integer.parseInt(param);
        String [] tablica = new String [ile];
        for (int i=0; i<ile; i++)
            tablica[i]=String.valueOf(i);
        request.setAttribute("tablica", tablica);
    } finally {
        out.close();
    }
}
request.getRequestDispatcher("/wyjscie.jsp").forward(request,
response);

```

Komunikacja serwletów

```

index.jsp:
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<form action="gen.do">
<input type="text" name="pole1"/>
<input type="submit" value="generuj" />
</form>
</body>
</html>

```

Komunikacja serwletów

```

wyjscie.jsp:
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>JSP Page</title>
</head>
<body>
<p>
<%
String [] tab = (String []) request.getAttribute("tablica");
String odp="";
for (int i=0; i<tab.length; i++)
    odp=odp+tab[i]+"<br>";
%>

<%= odp %>

</p>
</body>
</html>

```