Aplikacja internetowa zbudowana w oparciu o środowisko Visual Web Java Server Faces. Zarządzanie obiektami typu SesionBeans, RequestBeen i ApplicationBeans,

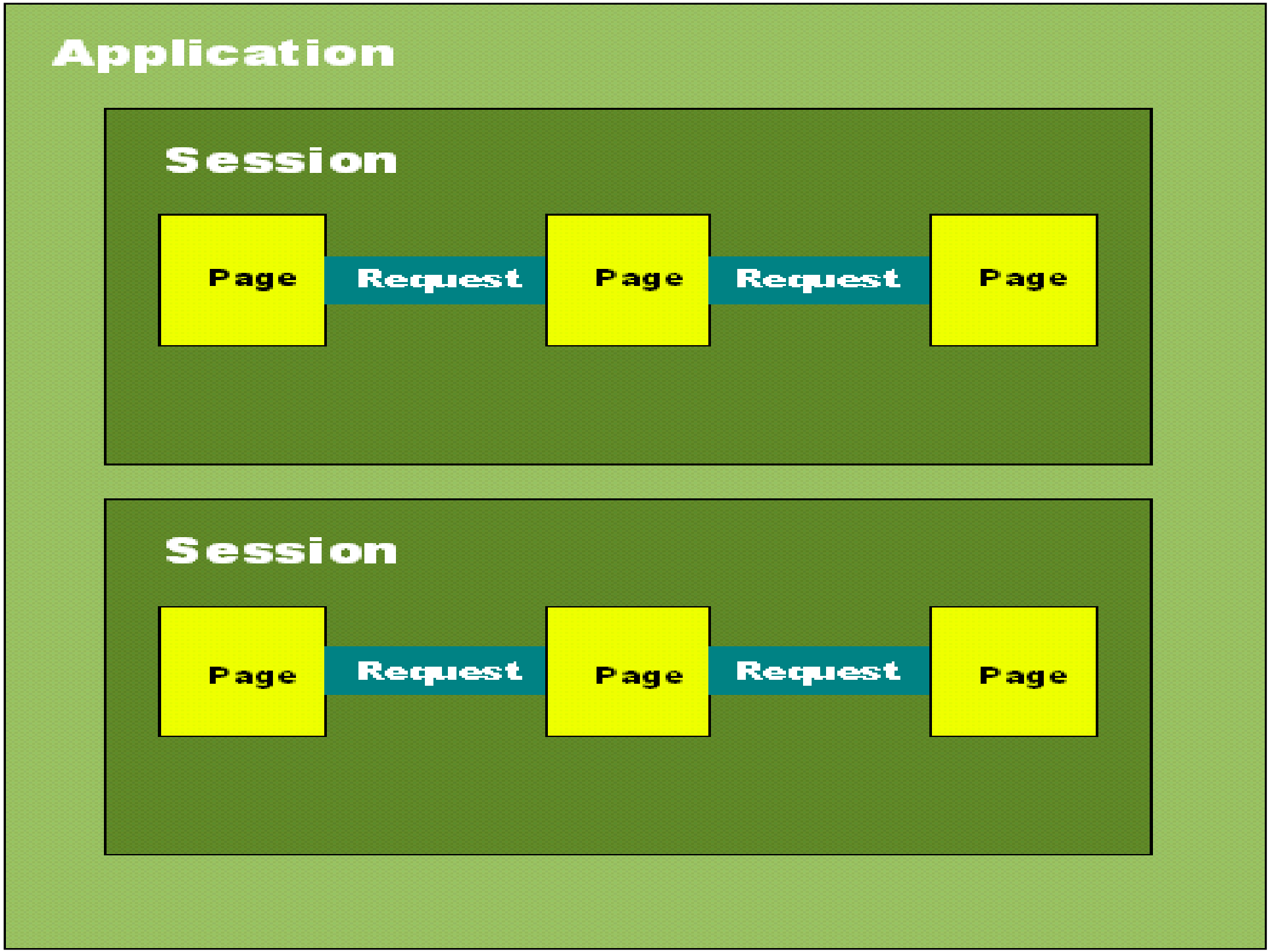
Laboratorium 1

Wzorce oprogramowania – lab1,
Zofia Kruczkiewicz

Okres życia obiektów

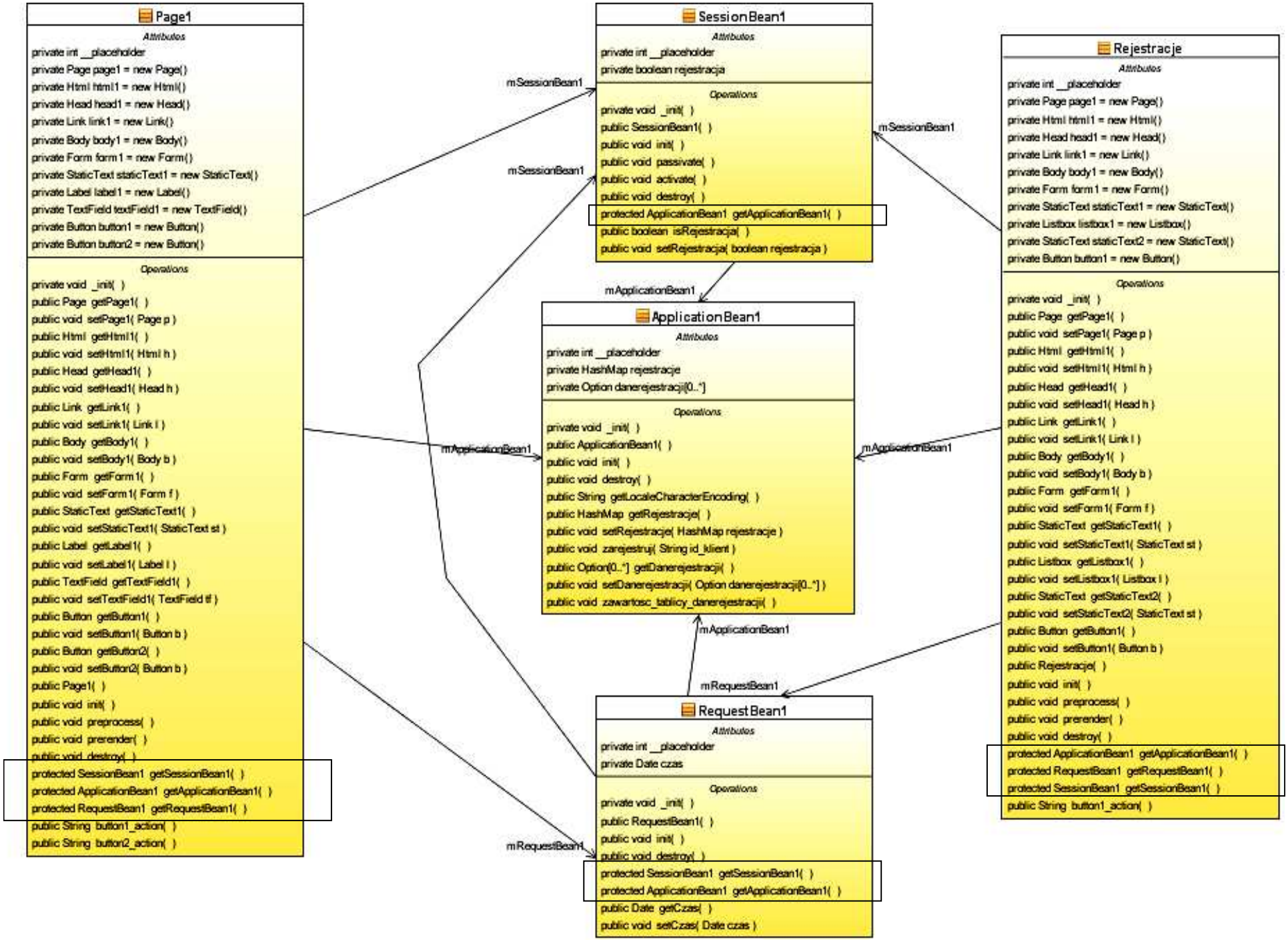
- Okres aplikacji oznacza czas życia obiektu typu ApplicationBean1 i kończy się, gdy serwer kończy wykonanie aplikacji. Wartości przechowywane w obiekcie typu ApplicationBean1 są dostępne przez cały czas życia każdej sesji i każdej fazy request należących do tej samej aplikacji.
- Okres sesji oznacza czas życia obiektu typu SessionBean1i zaczyna się, kiedy użytkownik po raz pierwszy wywołuje stronę aplikacji internetowej i kończy się, kiedy czas sesji kończy się zgodnie z wyznaczonym czasem typu Timeout lub gdy aplikacja internetowa przerywa sesję z powodu jej unieważnienia np. za pomocą metody session.invalidate().
- Okres żądań (okres request) oznacza czas życia obiektu typu RequestBean1 zaczyna się, kiedy użytkownik wysyła dane z formularza strony internetowej i kończy się, kiedy odpowiedź (faza response) jest w pełni zrealizowana.

  Ostrzeżenie: Nie można użyć obiektu typu RequestBean1, jeżeli strona zawiera element <redirect> wewnątrz elementu <navigation-case> w regule nawigacji. (Te reguły są widoczne, kiedy kliknie się przycisk XML w edytorze nawigacji strony (Page Navigation editor). Kiedy strona jest zatwierdzona (submit), element <redirect> przekierowuje do innej strony i kończy fazę request zanim osiągnięta nowa strona może wykorzystać wartości przechowywane w obiekcie typu RequestBean1.


Application
Session
Page
Request
Page
Request
Page
Session
Page
Request
Page
Request
Page

Przykład demonstrujący czas życia obiektów typu ApplicationBean1, SessionBean1 oraz RequestBean1

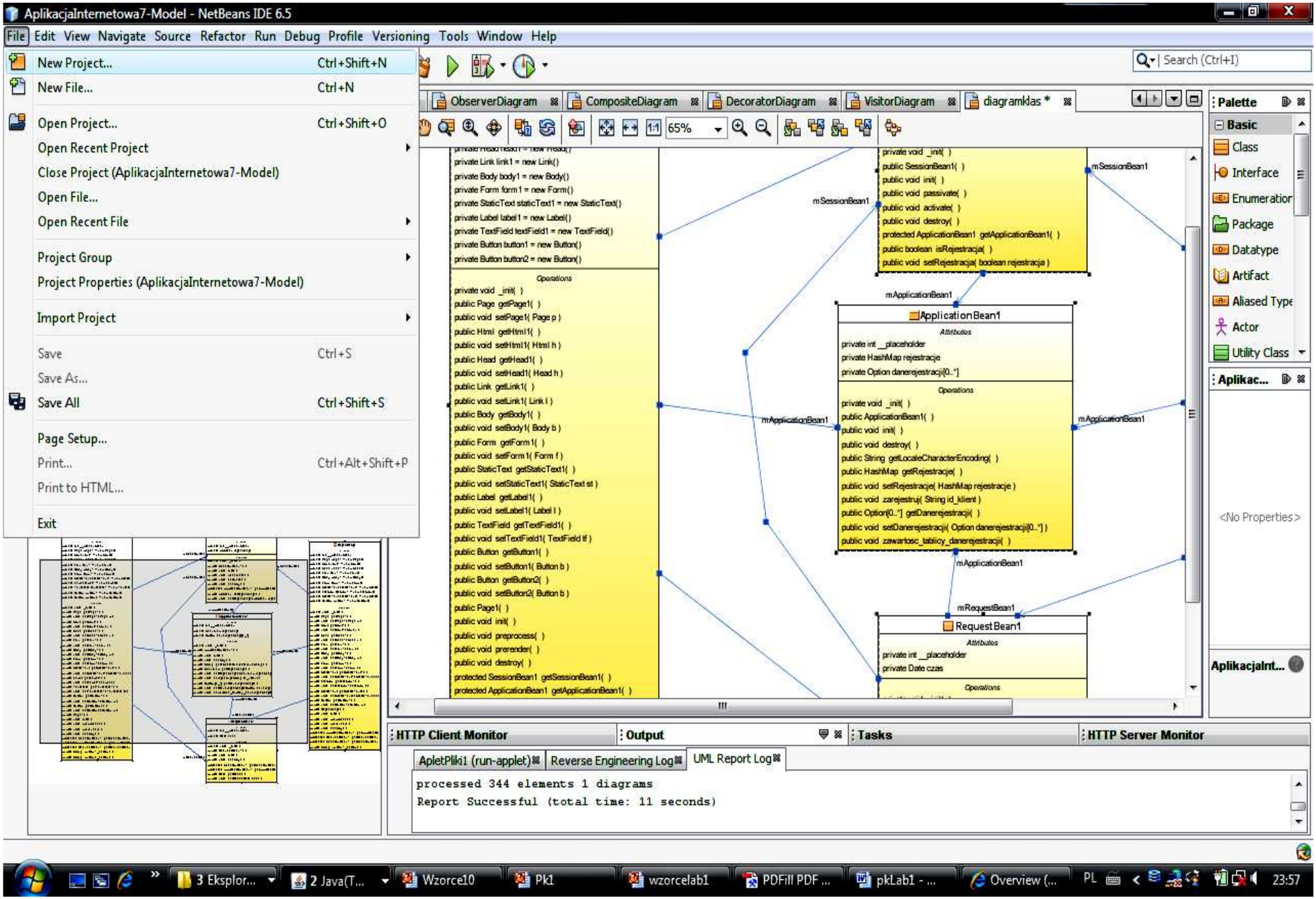
- **rejestracje.** Jest to pojemnik typu HashMap umieszczony w obiekcie typu ApplicationBean1, który przechowuje dane wszystkich rejestracji klientów tworzonych podczas kolejnego stanu sesji danego klienta
- **danerejestracji.** Jest to tablica elementów typu Option, umieszczona w obiekcie typu ApplicationBean1, do przechowania danych rejestracji w formie umożliwiającej wyświetlenie danych w komponencie typu List Box
- **rejestracja.** Jest to wartość typu logicznego tworzona podczas stanu sesji aplikacji, przechowywana w bieżącym obiekcie typu SessionBean1. Przyjmuje on wartość false, kiedy tworzy się nowy obiekt typu SessionBean1. Wartość true tego atrybutu jest ustawiana przy pierwszej rejestracji klienta podczas bieżącej sesji. Wartość true tego atrybutu wykorzystano do zablokowania ponownego rejestrowania się klienta w czasie jednej sesji
- **czas.** Czas ten jest odczytywany z czasu systemowego i przechowywany w bieżącym obiekcie typu RequestBean1
  - podczas rejestracji nowego klienta w obiekcie rejestracje i
  - żądania połączenia ze stroną do prezentowania danych zarejestrowanych klientów z obiektu danerejestracji oraz czasu bieżącej rejestracji z atrybutu czas


Page1
Attributes
private int __placeholder
private Page page1 = new Page()
private Html html1 = new Html()
private Head head1 = new Head()
private Link link1 = new Link()
private Body body1 = new Body()
private Form form1 = new Form()
private StaticText staticText1 = new StaticText()
private Label label1 = new Label()
private TextField textField1 = new TextField()
private Button button1 = new Button()
private Button button2 = new Button()
Operations
private void _init( )
public Page getPage1( )
public void setPage1(Page p)
public Html getHtml1( )
public void setHtml1(Html h)
public Head getHead1( )
public void setHead1(Head h)
public Link getLink1( )
public void setLink1(Link l)
public Body getBody1( )
public void setBody1(Body b)
public Form getForm1( )
public void setForm1(Form f)
public StaticText getStaticText1( )
public void setStaticText1(StaticText st)
public Label getLabel1( )
public void setLabel1(Label l)
public TextField getTextField1( )
public void setTextField1(TextField tf)
public Button getButton1( )
public void setButton1(Button b)
public Button getButton2( )
public void setButton2(Button b)
public Page1( )
public void init( )
public void preprocess( )
public void prerender( )
public void destroy( )
protected SessionBean1 getSessionBean1( )
protected ApplicationBean1 getApplicationBean1( )
protected RequestBean1 getRequestBean1( )
public String button1_action( )
public String button2_action( )
SessionBean1
Attributes
private int __placeholder
private boolean rejestracja
Operations
private void _init( )
public SessionBean1( )
public void init( )
public void passivate( )
public void activate( )
public void destroy( )
protected ApplicationBean1 getApplicationBean1( )
public boolean isRejestracja( )
public void setRejestracja(boolean rejestracja)
Rejestracje
Attributes
private int __placeholder
private Page page1 = new Page()
private Html html1 = new Html()
private Head head1 = new Head()
private Link link1 = new Link()
private Body body1 = new Body()
private Form form1 = new Form()
private StaticText staticText1 = new StaticText()
private Listbox listbox1 = new Listbox()
private StaticText staticText2 = new StaticText()
private Button button1 = new Button()
Operations
private void _init( )
public Page getPage1( )
public void setPage1(Page p)
public Html getHtml1( )
public void setHtml1(Html h)
public Head getHead1( )
public void setHead1(Head h)
public Link getLink1( )
public void setLink1(Link l)
public Body getBody1( )
public void setBody1(Body b)
public Form getForm1( )
public void setForm1(Form f)
public StaticText getStaticText1( )
public void setStaticText1(StaticText st)
public Listbox getListbox1( )
public void setListbox1(Listbox l)
public StaticText getStaticText2( )
public void setStaticText2(StaticText st)
public Button getButton1( )
public void setButton1(Button b)
public Rejestracje( )
public void init( )
public void preprocess( )
public void prerender( )
public void destroy( )
protected ApplicationBean1 getApplicationBean1( )
protected RequestBean1 getRequestBean1( )
protected SessionBean1 getSessionBean1( )
public String button1_action( )
ApplicationBean1
Attributes
private int __placeholder
private HashMap rejestracje
private Option danerejestracji[0..*]
Operations
private void _init( )
public ApplicationBean1( )
public void init( )
public void destroy( )
public String getLocaleCharacterEncoding( )
public HashMap getRejestracje( )
public void setRejestracje(HashMap rejestracje)
public void zarejestruj(String id_klient)
public Option[0..*] getDanerejestracji( )
public void setDanerejestracji(Option danerejestracji[0..*])
public void zawartosc_tablicy_danerejestracji( )
RequestBean1
Attributes
private int __placeholder
private Date czas
Operations
private void _init( )
public RequestBean1( )
public void init( )
public void destroy( )
protected SessionBean1 getSessionBean1( )
protected ApplicationBean1 getApplicationBean1( )
public Date getCzas( )
public void setCzas(Date czas)
mSessionBean1
mSessionBean1
mSessionBean1
mApplicationBean1
mApplicationBean1
mApplicationBean1
mApplicationBean1
mRequestBean1
mRequestBean1

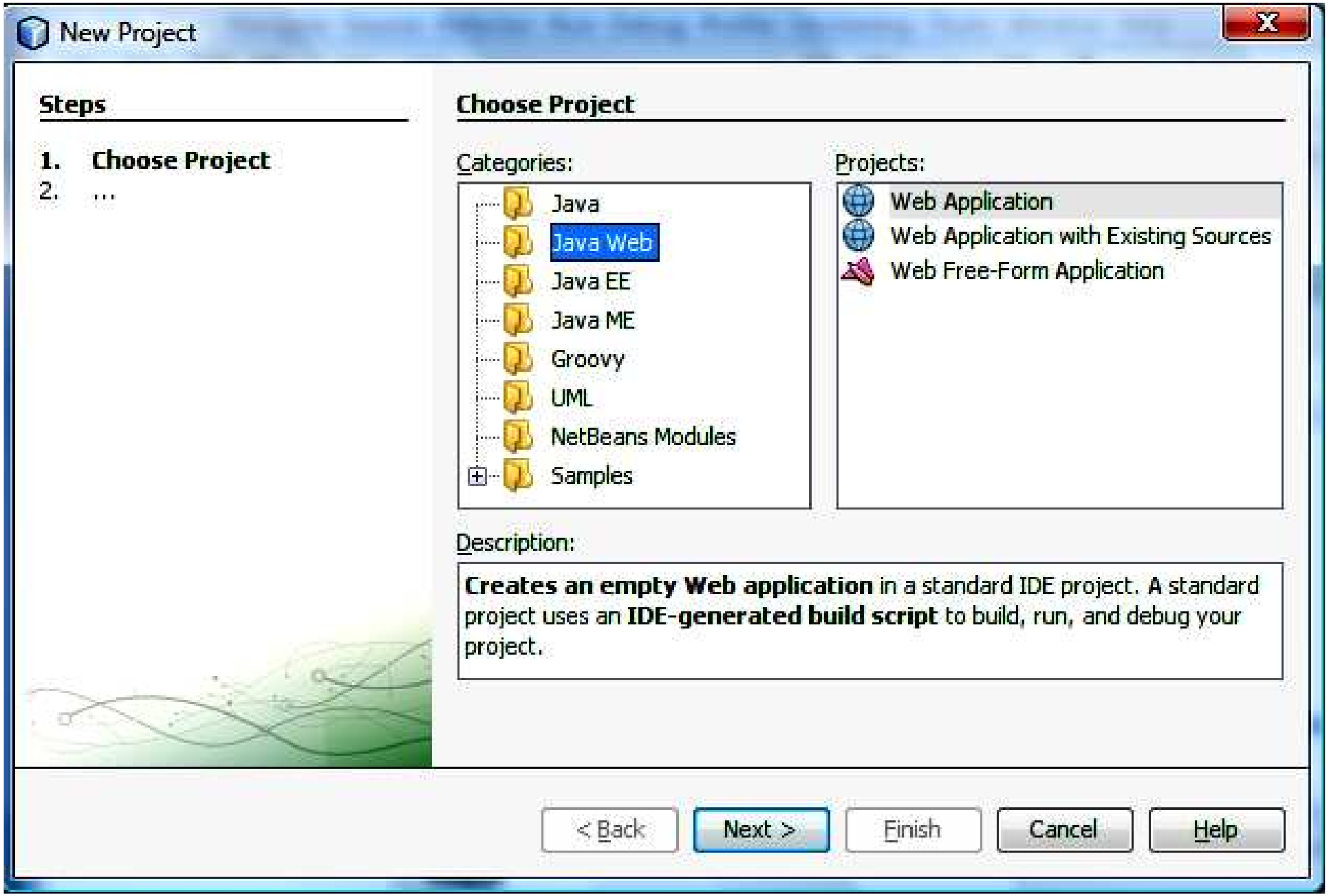
1. Informacje dotyczące realizacji programu w środowisku NetBeans 6.5

1) Z głównego menu wybierz opcję File > New Project.
2) W New Project, wybierz Java Web listy kategorii (Categories) i wybierz Web Application z listy projektów (Projects). Kliknij Next.
3) Nazwij projekt AplikacjaInternetowa7 (Project Name) i wybierz katalog (Browse dla Project Location). Kliknij Next.
4) Wybierz serwer aplikacji GlassFish V2 z listy Server oraz wersję Java EE z listy Java EE Version – domyślne wartości. Kliknij na Next
5) Wybierz Visual Web JavaServer Faces i naciśnij Finish.
6) Wybierz okno projektu (zakładka Projects) - zawiera ono układ plików typu BluePrints. Plik Page1.jsp jest stroną startową napisaną w języku JSP (zaznaczenie zielonym poziomym trójkątem) i znajduje się w podkatalogu „Web Pages”.

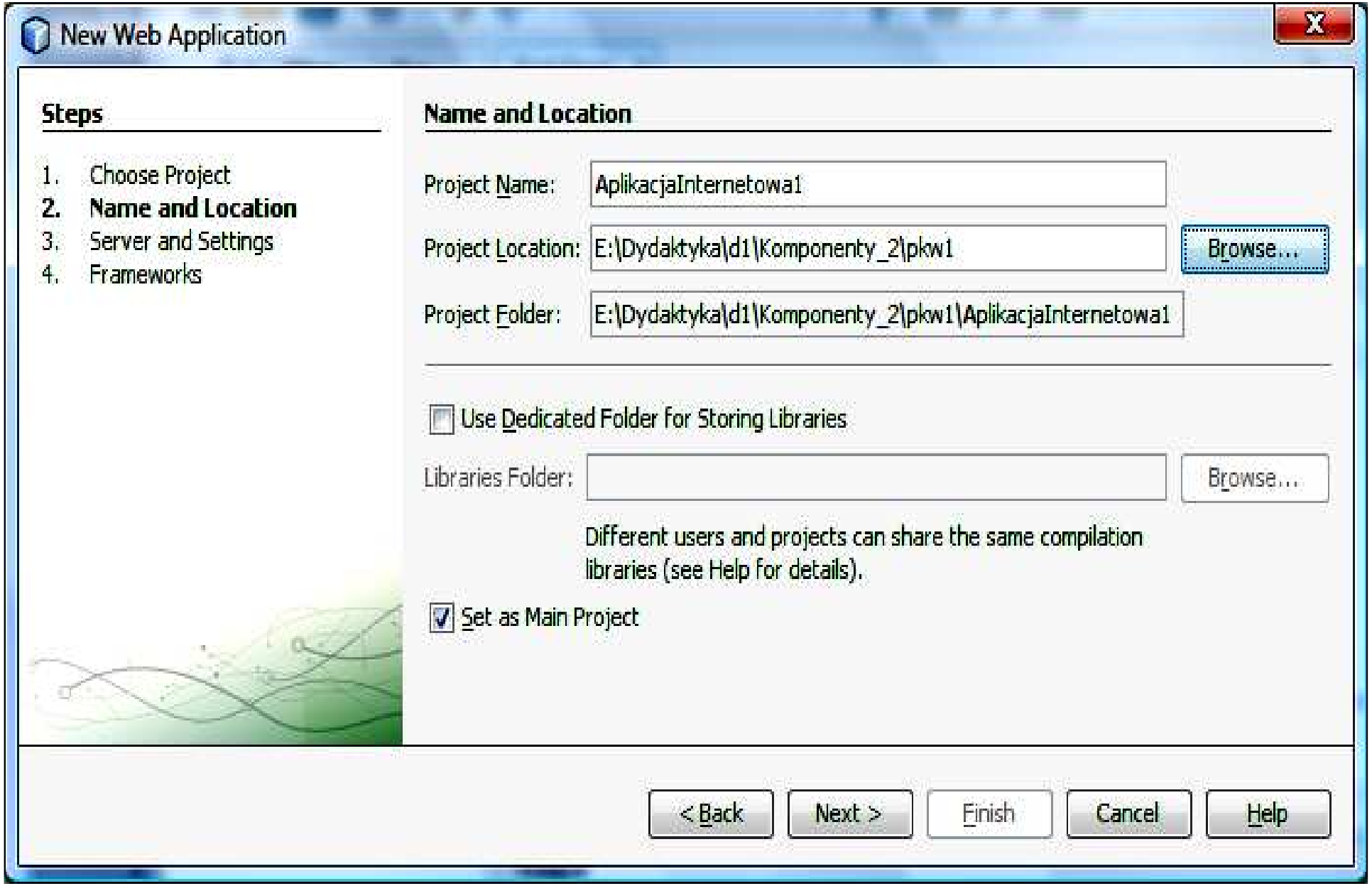
Ad. 1)


AplikacjaInternetowa7-Model - NetBeans IDE 6.5
File Edit View Navigate Source Refactor Run Debug Profile Versioning Tools Window Help
New Project... Ctrl+Shift+N
New File... Ctrl+N
Open Project... Ctrl+Shift+O
Open Recent Project
Close Project (AplikacjaInternetowa7-Model)
Open File...
Open Recent File
Project Group
Project Properties (AplikacjaInternetowa7-Model)
Import Project
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Page Setup...
Print... Ctrl+Alt+Shift+P
Print to HTML...
Exit
ObserverDiagram
CompositeDiagram
DecoratorDiagram
VisitorDiagram
diagramklas *
Palette
Basic
Class
Interface
Enumeration
Package
Datatype
Artifact
Aliased Type
Actor
Utility Class
ApplicationBean1
RequestBean1
mSessionBean1
mApplicationBean1
mRequestBean1
HTTP Client Monitor
Output
Tasks
HTTP Server Monitor
ApletPlik1 (run-applet)
Reverse Engineering Log
UML Report Log
processed 344 elements 1 diagrams
Report Successful (total time: 11 seconds)

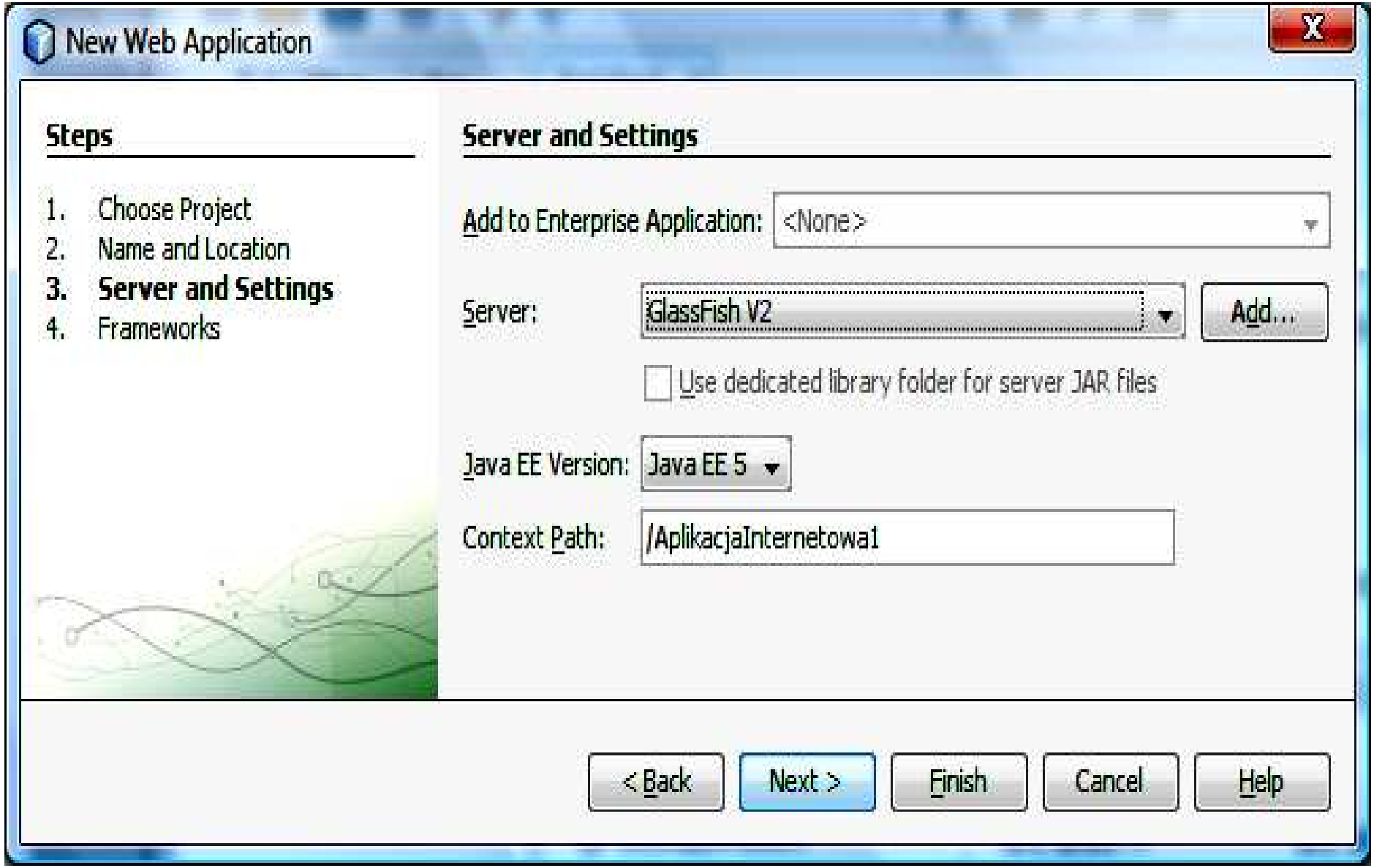
Ad. 2)


New Project
Steps
1. Choose Project
2. ...
Choose Project
Categories:
Java
Java Web
Java EE
Java ME
Groovy
UML
NetBeans Modules
Samples
Projects:
Web Application
Web Application with Existing Sources
Web Free-Form Application
Description:
Creates an empty Web application in a standard IDE project. A standard project uses an IDE-generated build script to build, run, and debug your project.
< Back
Next >
Finish
Cancel
Help

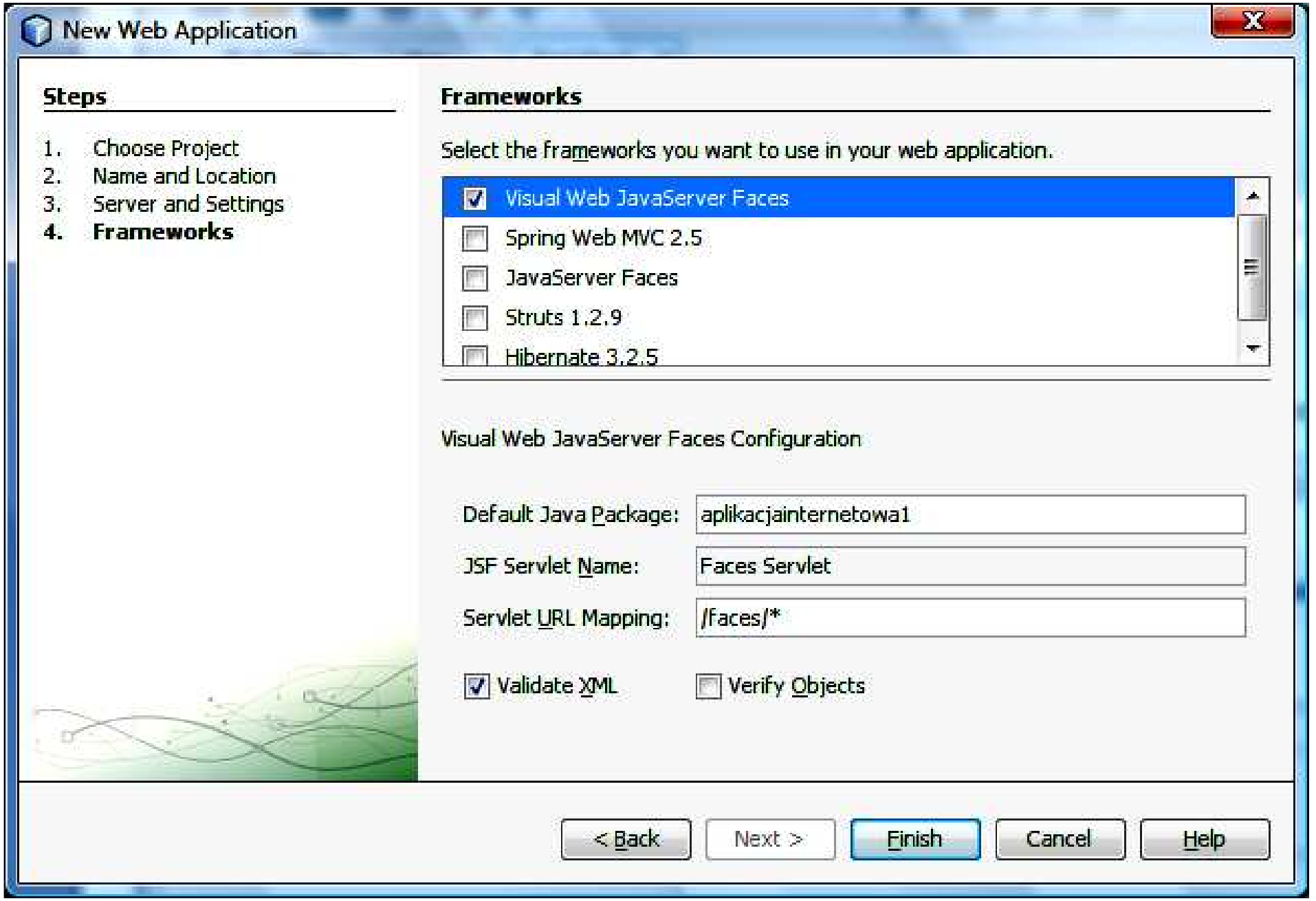
Ad. 3)


New Web Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
4. Frameworks
Name and Location
Project Name: AplikacjaInternetowa1
Project Location: E:\Dydaktyka\d1\Komponenty_2\pkw1
Browse...
Project Folder: E:\Dydaktyka\d1\Komponenty_2\pkw1\AplikacjaInternetowa1
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
Set as Main Project
< Back
Next >
Finish
Cancel
Help

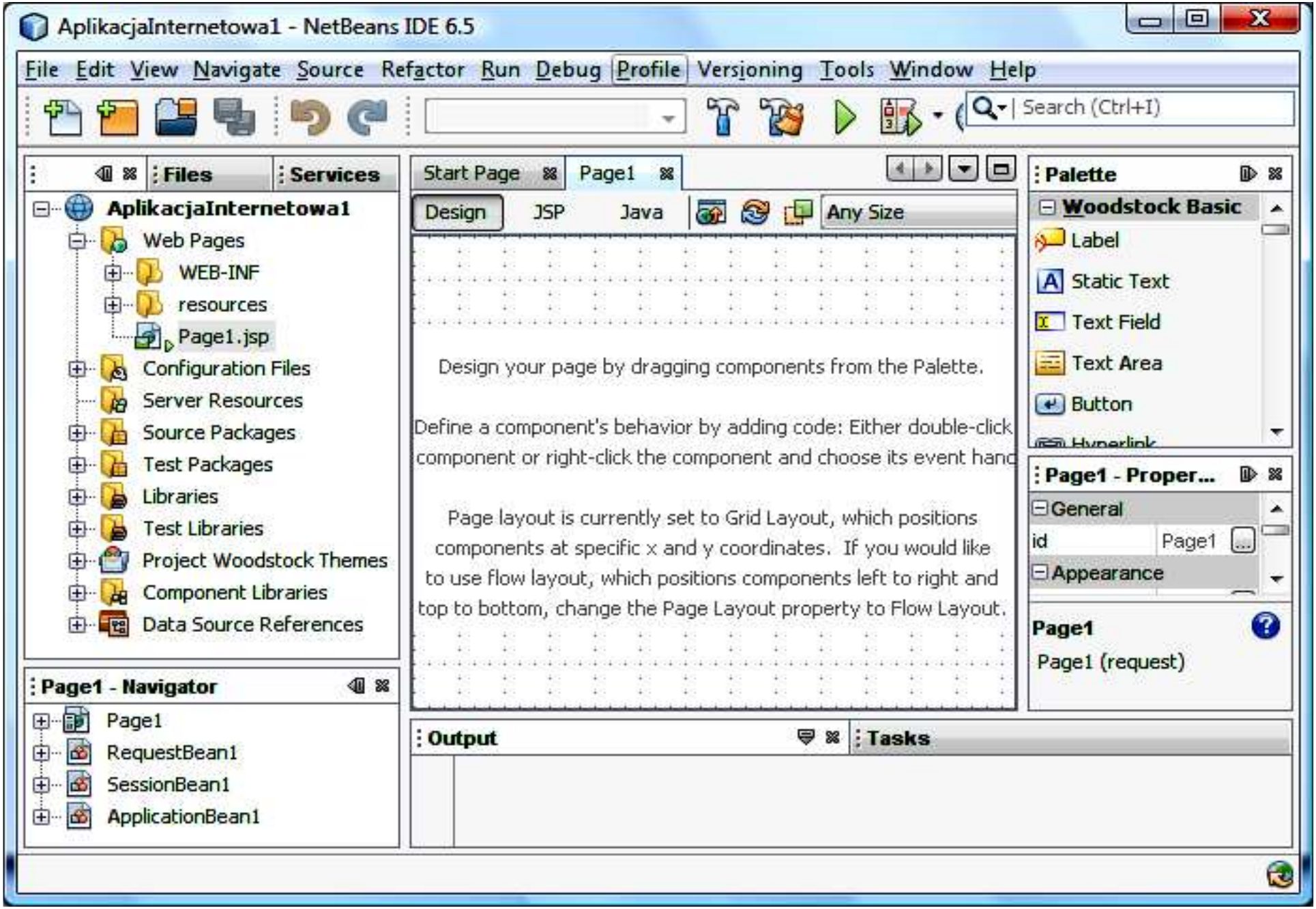
Ad. 4)


New Web Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
4. Frameworks
Server and Settings
Add to Enterprise Application: <None>
Server: GlassFish V2
Add...
Use dedicated library folder for server JAR files
Java EE Version: Java EE 5
Context Path: /AplikacjaInternetowa1
< Back
Next >
Finish
Cancel
Help

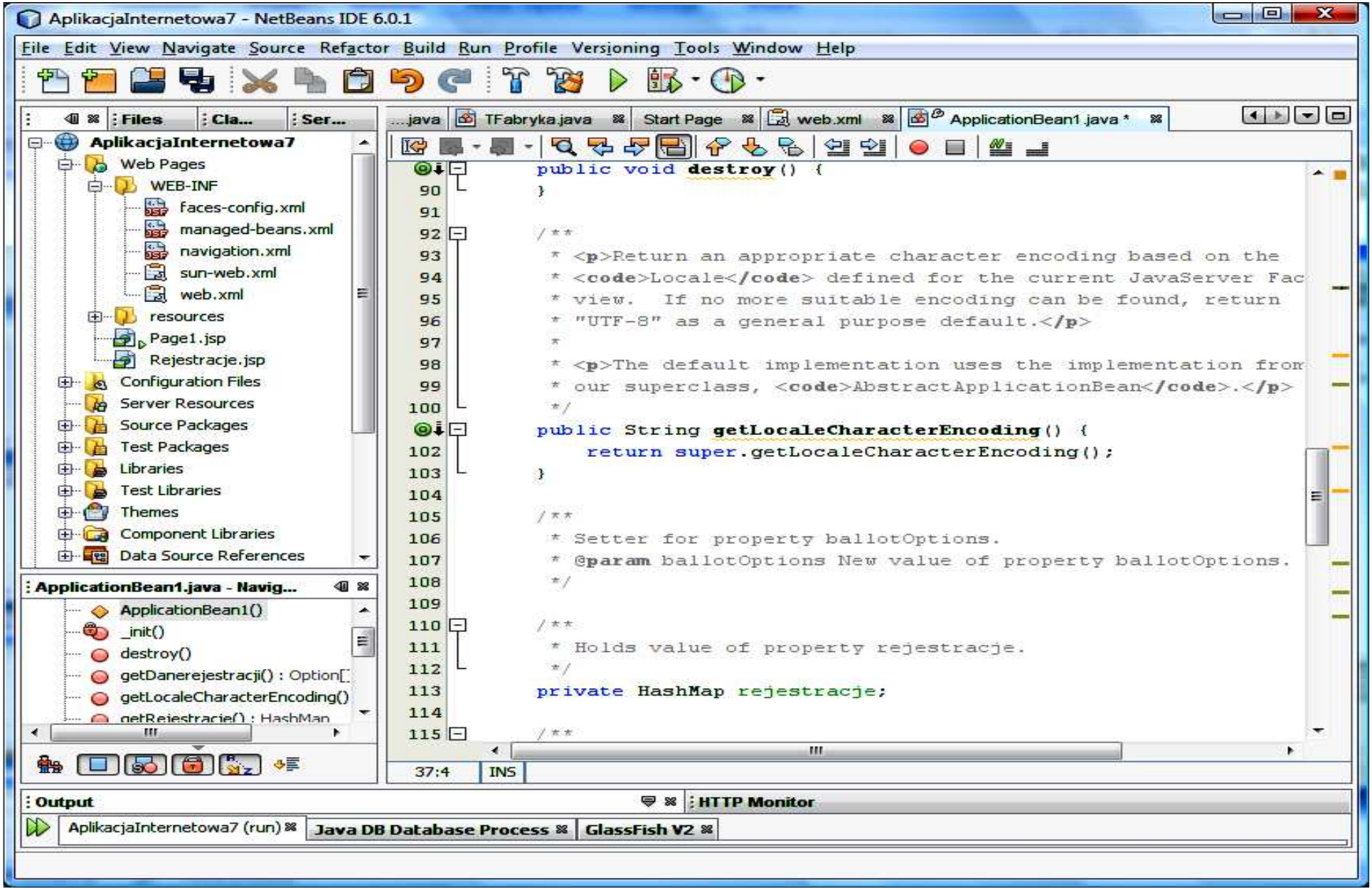
Ad. 5)


New Web Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
4. Frameworks
Frameworks
Select the frameworks you want to use in your web application.
Visual Web JavaServer Faces
Spring Web MVC 2.5
JavaServer Faces
Struts 1.2.9
Hibernate 3.2.5
Visual Web JavaServer Faces Configuration
Default Java Package:
aplikacjainternetowa1
JSF Servlet Name:
Faces Servlet
Servlet URL Mapping:
/faces/*
Validate XML
Verify Objects
< Back
Next >
Finish
Cancel
Help

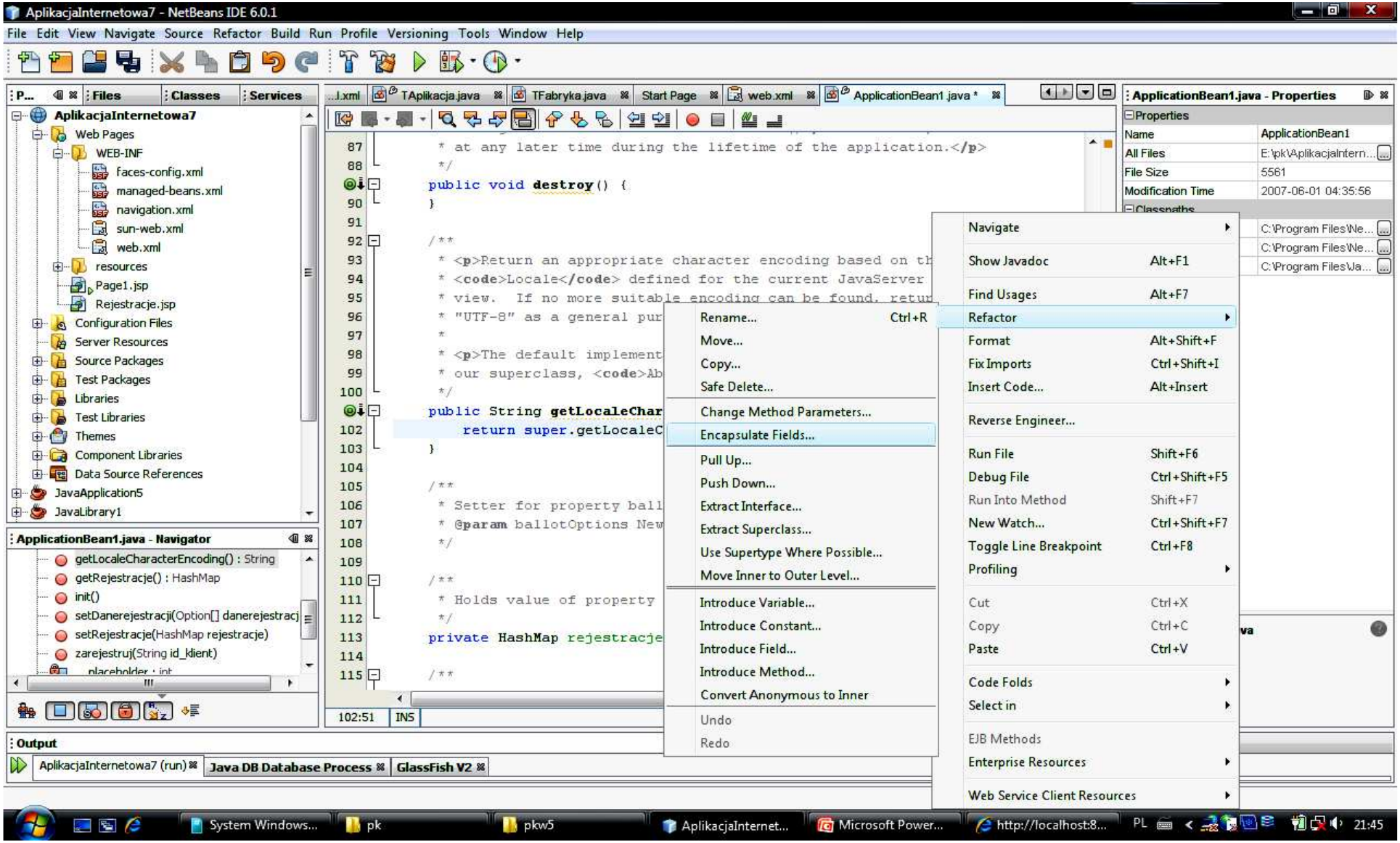
Ad. 6) Utworzenie aplikacji w Visual Web Pack – AplikacjaInternetowa7



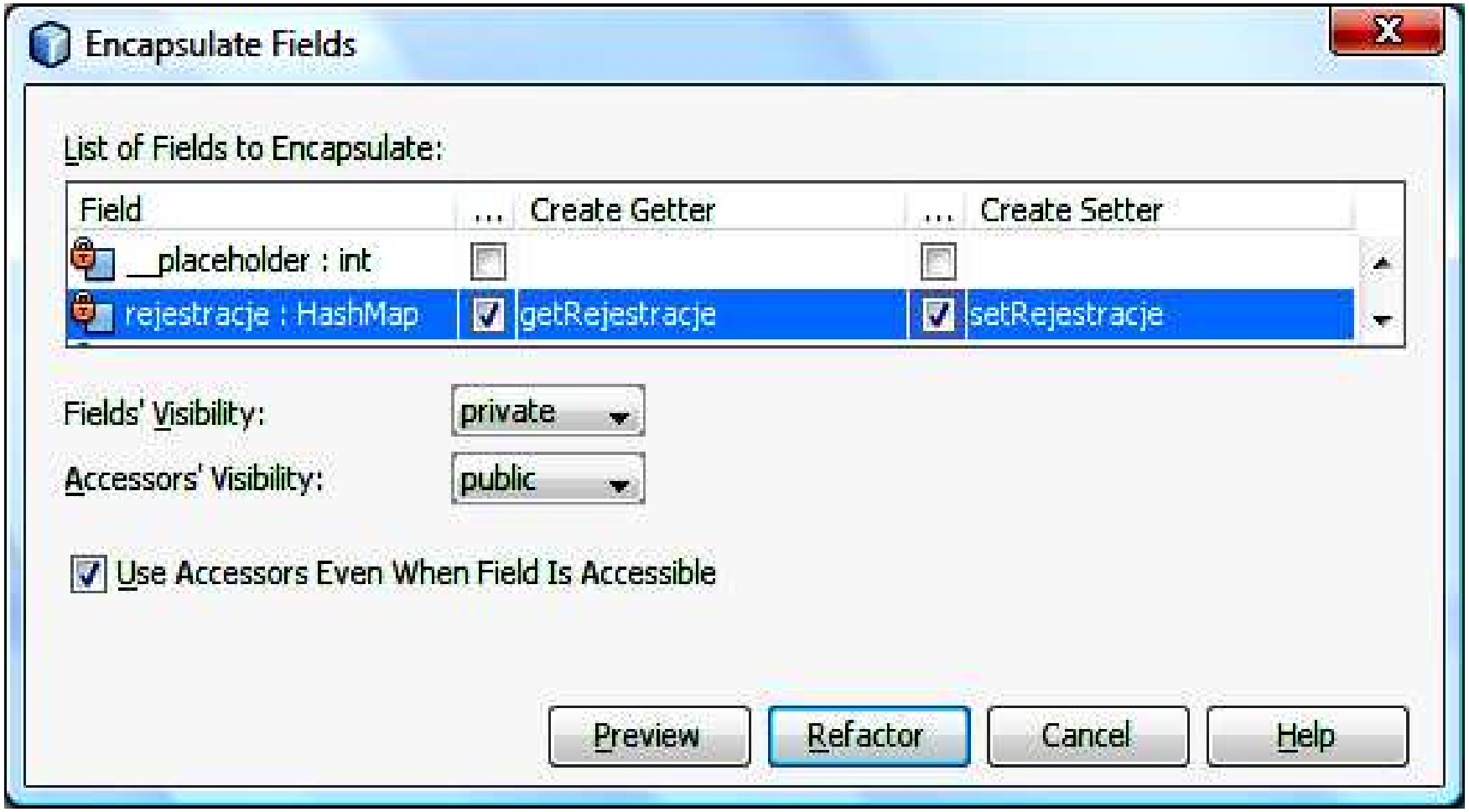
2. Dodawanie atrybutu rejestracje typu HashMap do klasy typu ApplicationBean1 – ręczne wpisanie linii kodu.
Import brakujących pakietów - klawisze CTRL+Shift+I



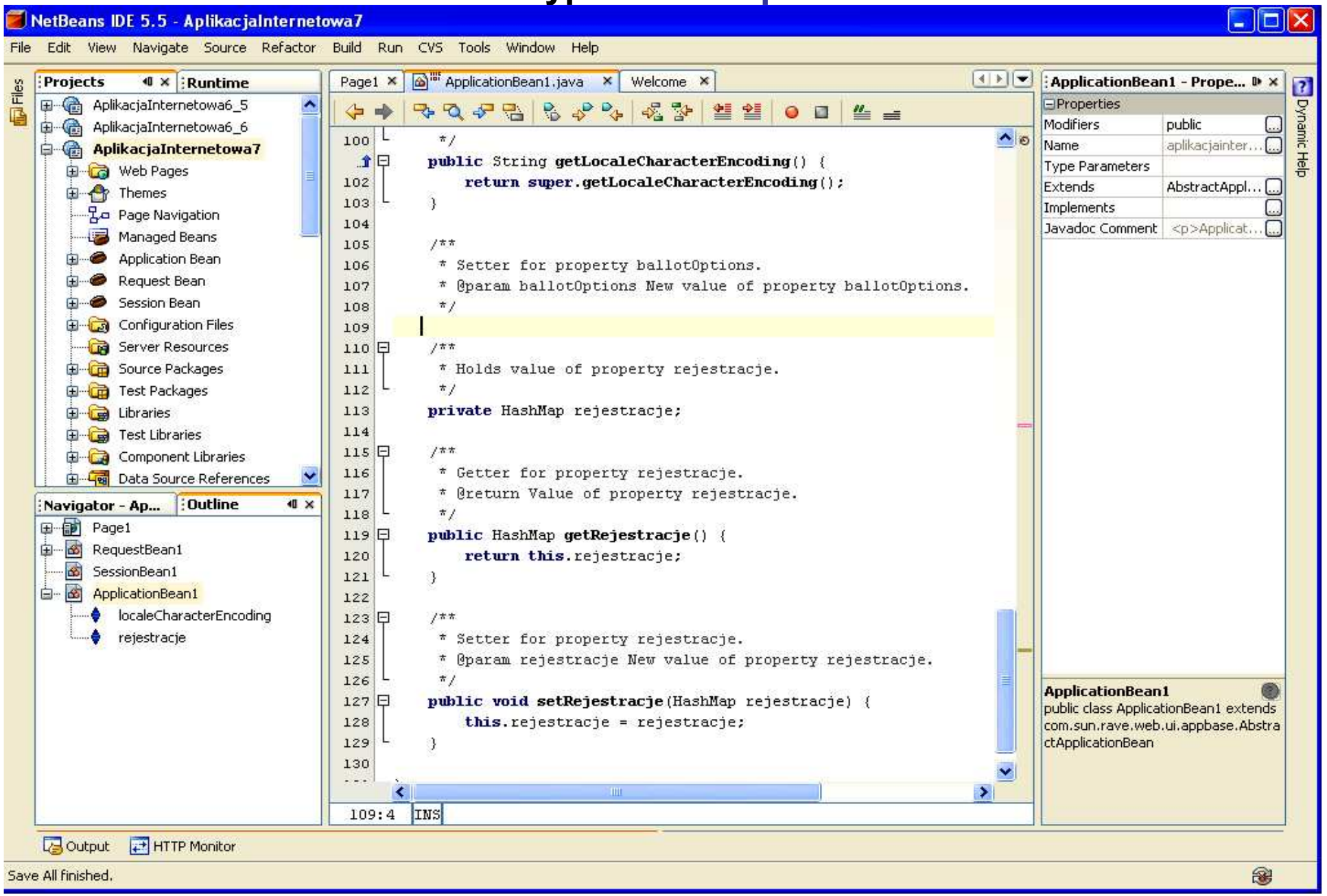
2.1. Dodanie metod typu set i get dla atrybutu rejestracje typu HashMap do klasy ApplicationBean1 w celu przechowania danych rejestrowanych klientów podczas każdej sesji aplikacji internetowej – kliknąć prawym klawiszem myszy na ciało klasy w trybie Java, wybrać Refactor, następnie Encapsulate Fields



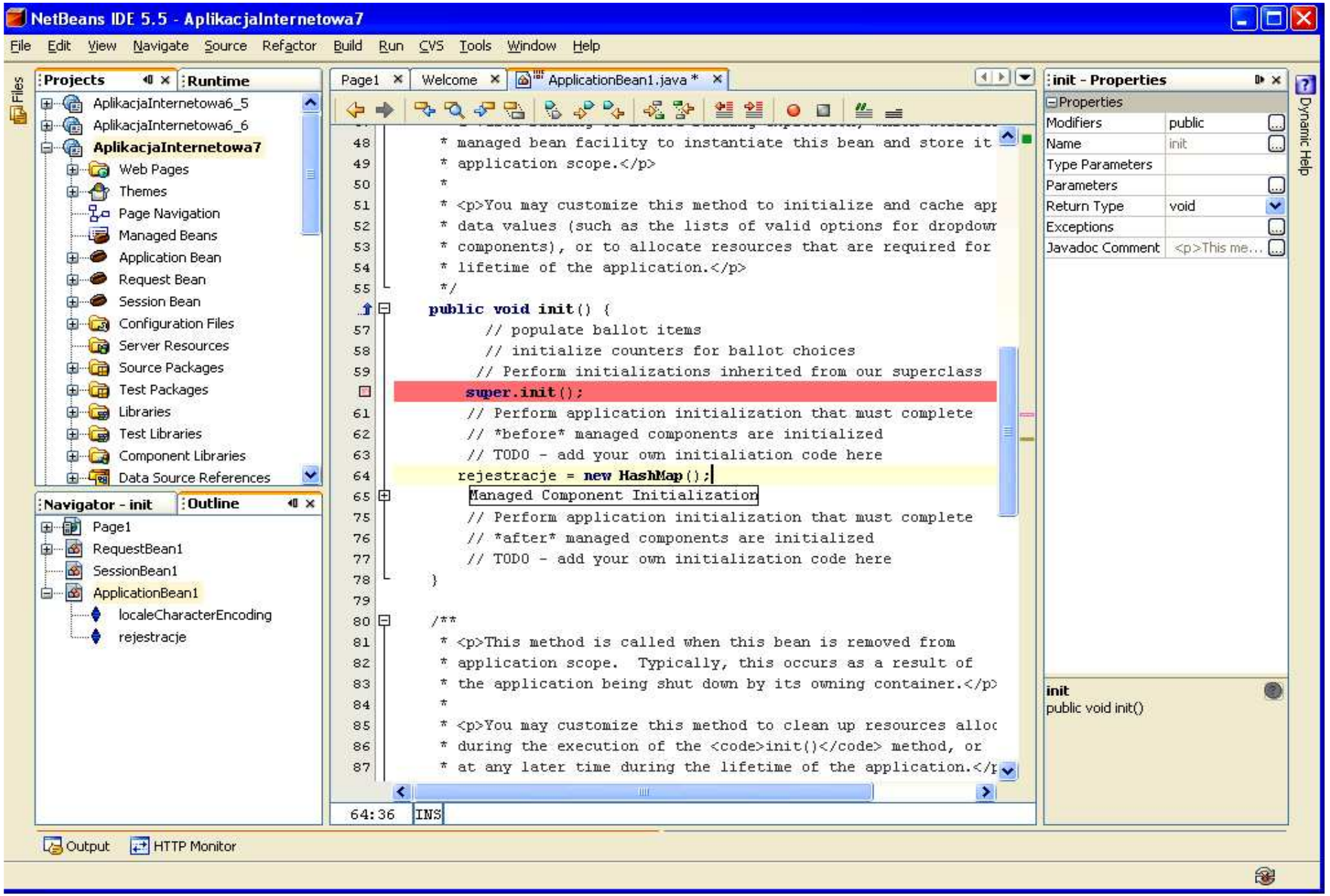
2.2. Dodanie metod typu set i get dla atrybutu rejestracje typu HashMap do klasy ApplicationBean1 w celu przechowania danych rejestrowanych klientów podczas każdej sesji aplikacji internetowej cd. – zaznaczyć Create Getter oraz Create Setter i nacisnąć Refactor



2.3. Rezultat działań – obiekt ApplicationBean1 posiada atrybut rejestracje typu HashMap.



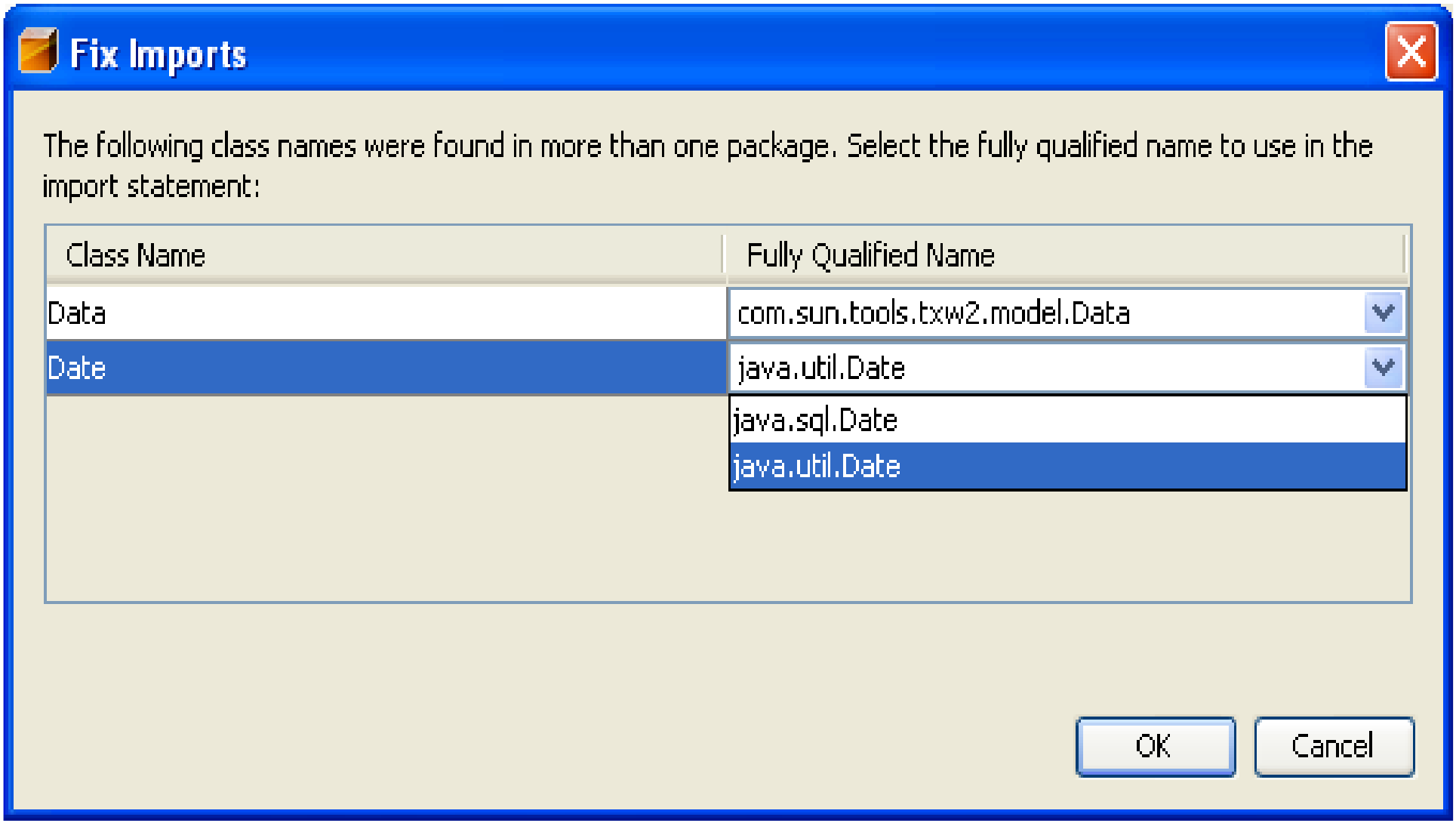
2.4. Uzupełnienie kodu metody init w klasie ApplicationBean1 – metoda init w każdym obiekcie uruchamiana jest w momencie tworzenia obiektu



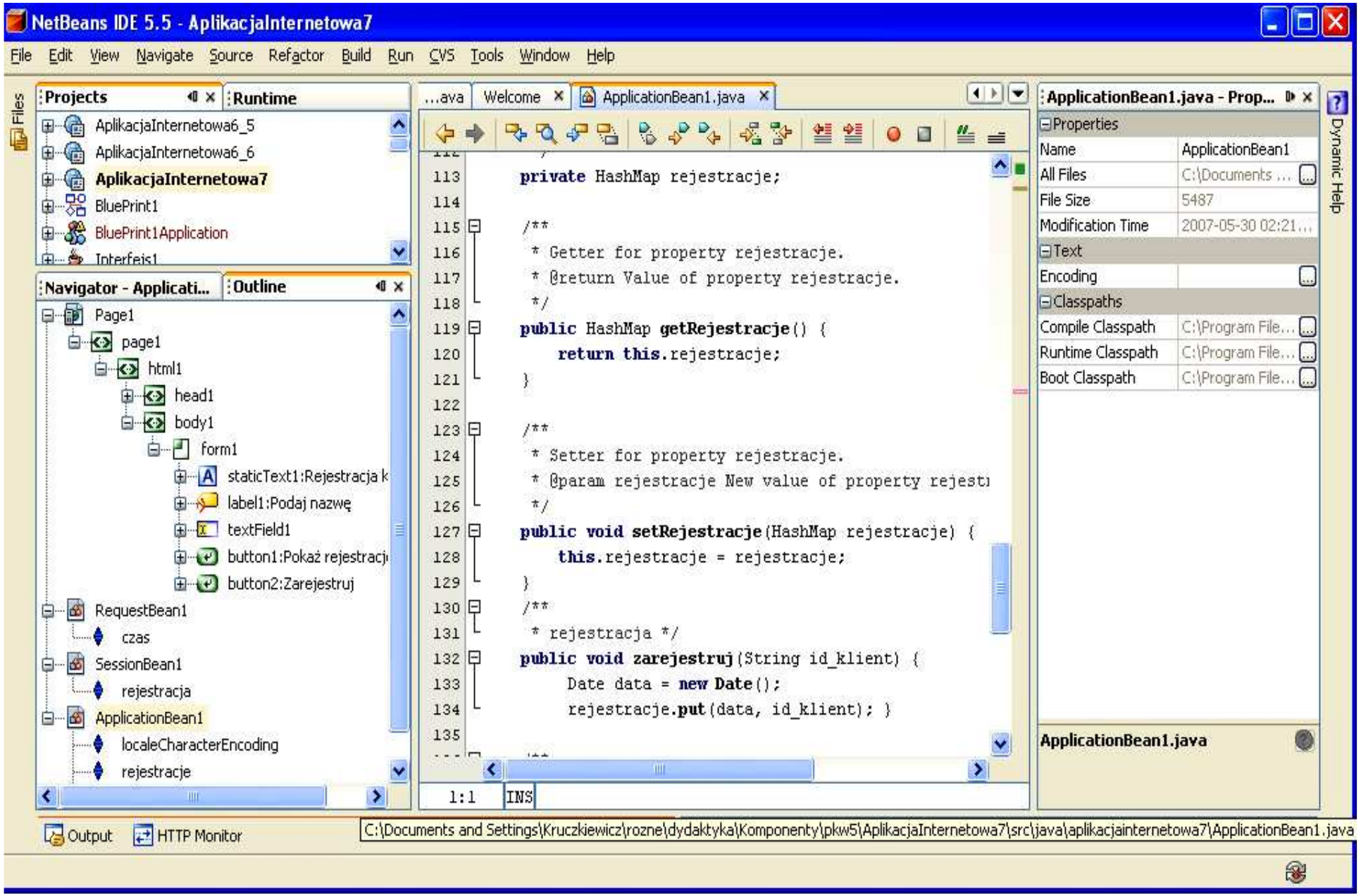
2.5. Definicja metody zarejestruj w klasie ApplicationBean1, która rejestruje klientów za pomocą pary danych wstawianych do kolekcji rejestracje typu HashMap wstawiając jako klucz datę bieżącą i dane, które stanowią identyfikator klienta

```
 /**
* rejestracja */
public void zarejestruj(String id_klient)
{
    Date data = new Date();
    rejestracje.put(data, id_klient);
}
```

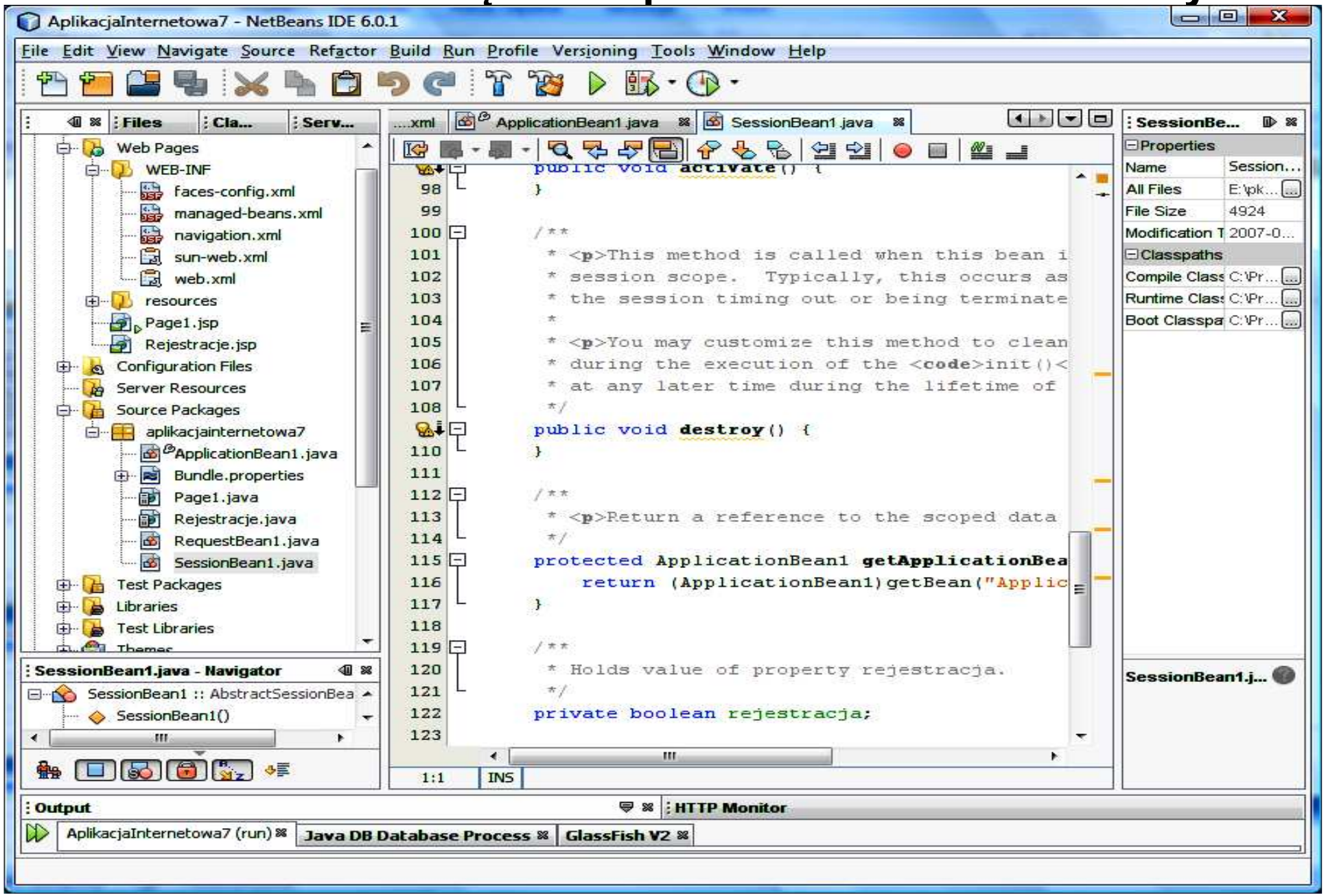
2.6. Import brakujących pakietów - klawisze CTRL+Shift+I oraz wybór w formularzu, który pokazuje się w sytuacji, kiedy należy wybrać właściwą klasę



2.7. Rezultat działań – obiekt ApplicationBean1 posiada metodę zarejestruj.



3. Dodawanie atrybutu rejestracja typu boolean do klasy typu SessionBean1 - ręczne wpisanie kodu do ciała klasy

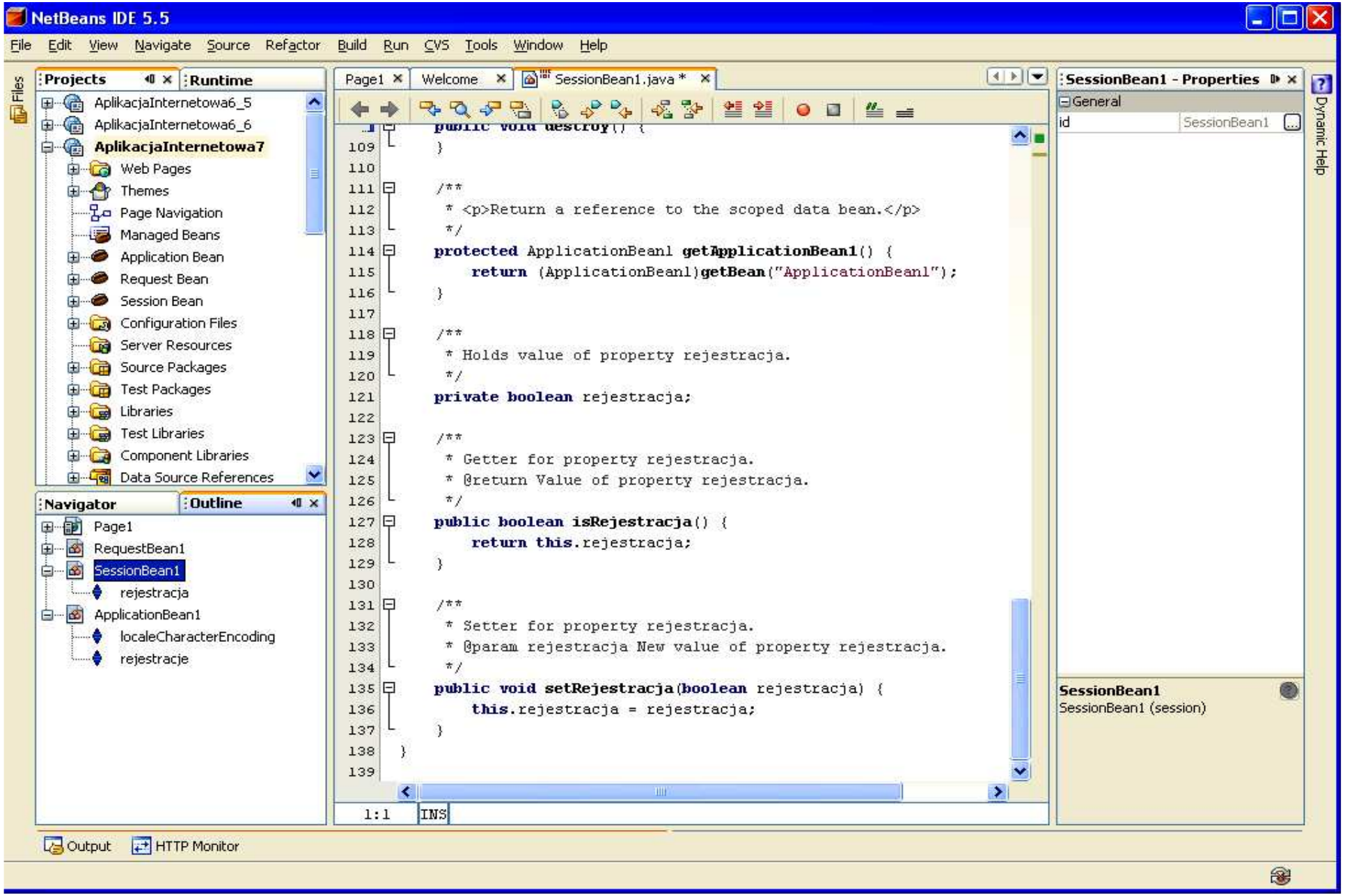


3.1. Dodanie metod typu set i get dla atrybutu rejestracja typu boolean do klasy SessionBean1 w celu w celu umożliwienia rejestracji klientowi podczas każdej sesji tylko raz – kliknąć prawym klawiszem myszy na ciało klasy w trybie Java, wybrać Refactor, następnie Encapsulate Fields.

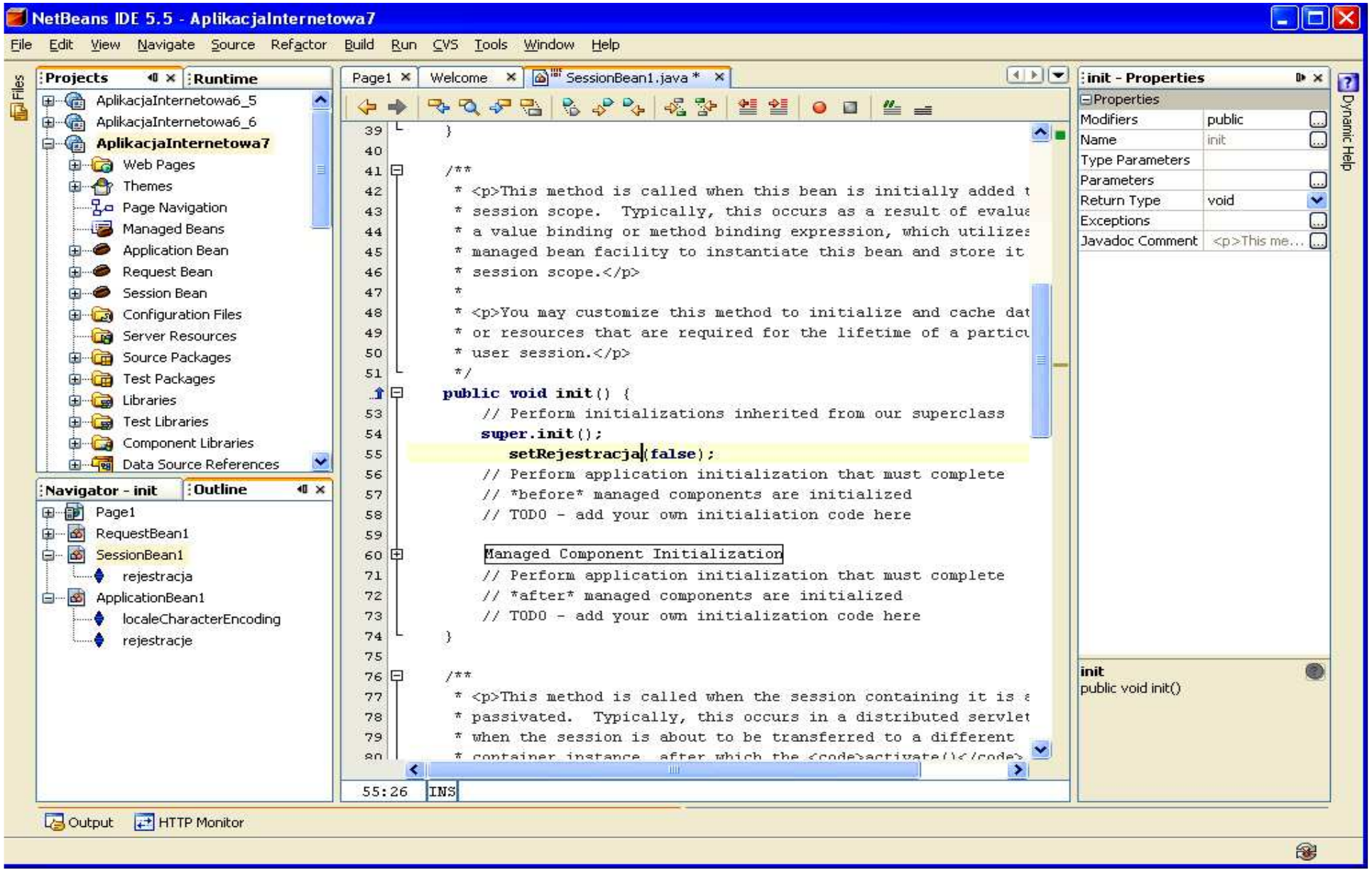
3.2. Dodanie metod typu set i get dla atrybutu rejestracja typu boolean do klasy SessionBean1 w celu umożliwienia rejestracji klientowi podczas każdej sesji tylko raz cd. – zaznaczyć Create Getter oraz Create Setter
i nacisnąć **Refactor**

Uwagi:
W momencie wywołania głównej strony aplikacji Page1 tworzony jest obiekt typu SessionBean1 i zmienna rejestracja przyjmuje wartość false. Podczas rejestracji klienta zmienna rejestracja przyjmuje wartość true, co blokuje ponowne wprowadzanie danych do rejestracji. Dopiero, kiedy obiekt typu SesionBean1 zostanie zniszczony (po czasie wyznaczonym przez TimeOut) i ponownie utworzony przy kolejnym wywołaniu strony internetowej, zmienna rejestracja przyjmuje wartość false. Klient może teraz zarejestrować się, ponieważ formularz rejestracji jest odblokowany dzięki wartości zmiennej rejestracja równej false.

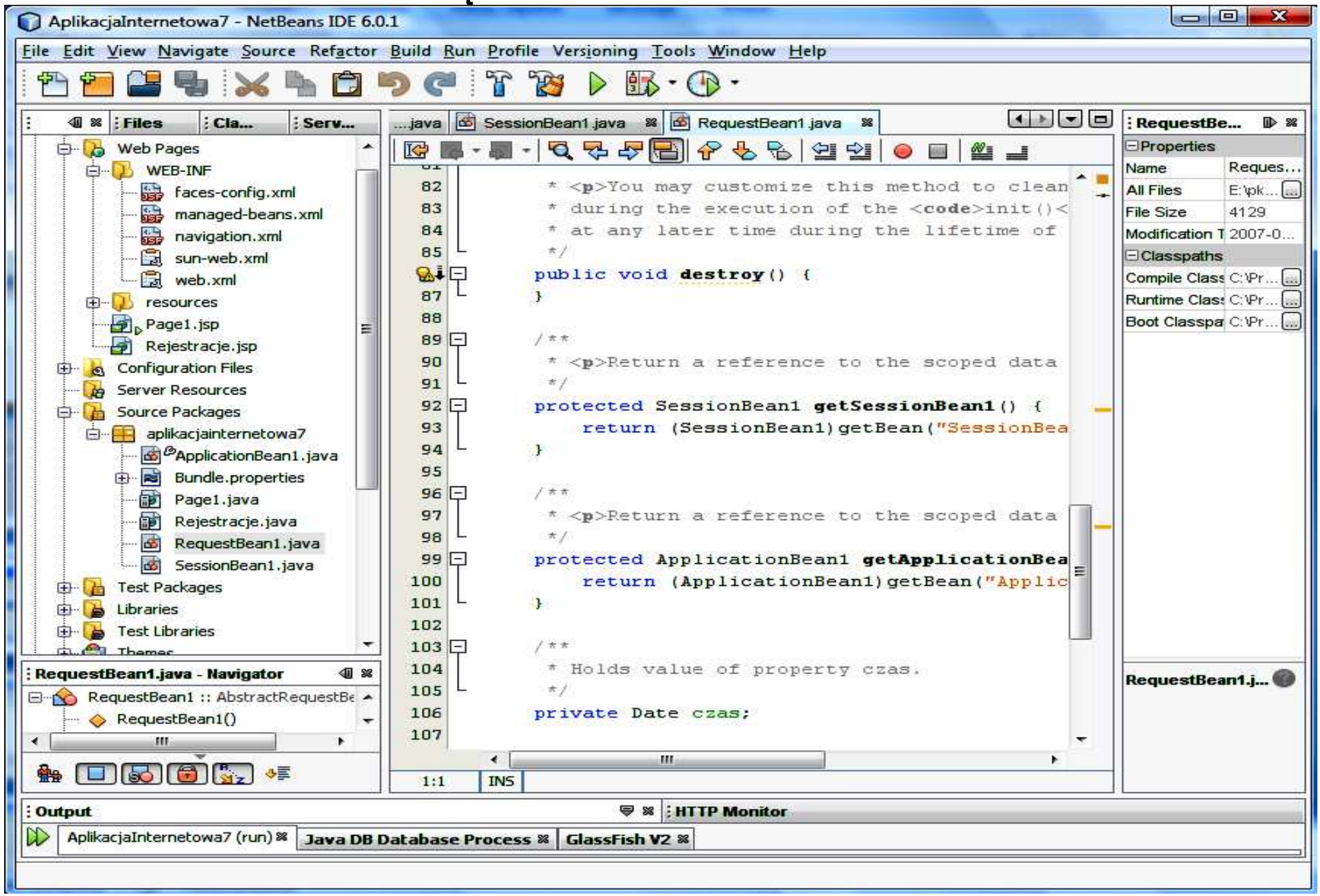
3.3. Rezultat działań – obiekt SessionBean1 posiada atrybut rejestracja typu boolean



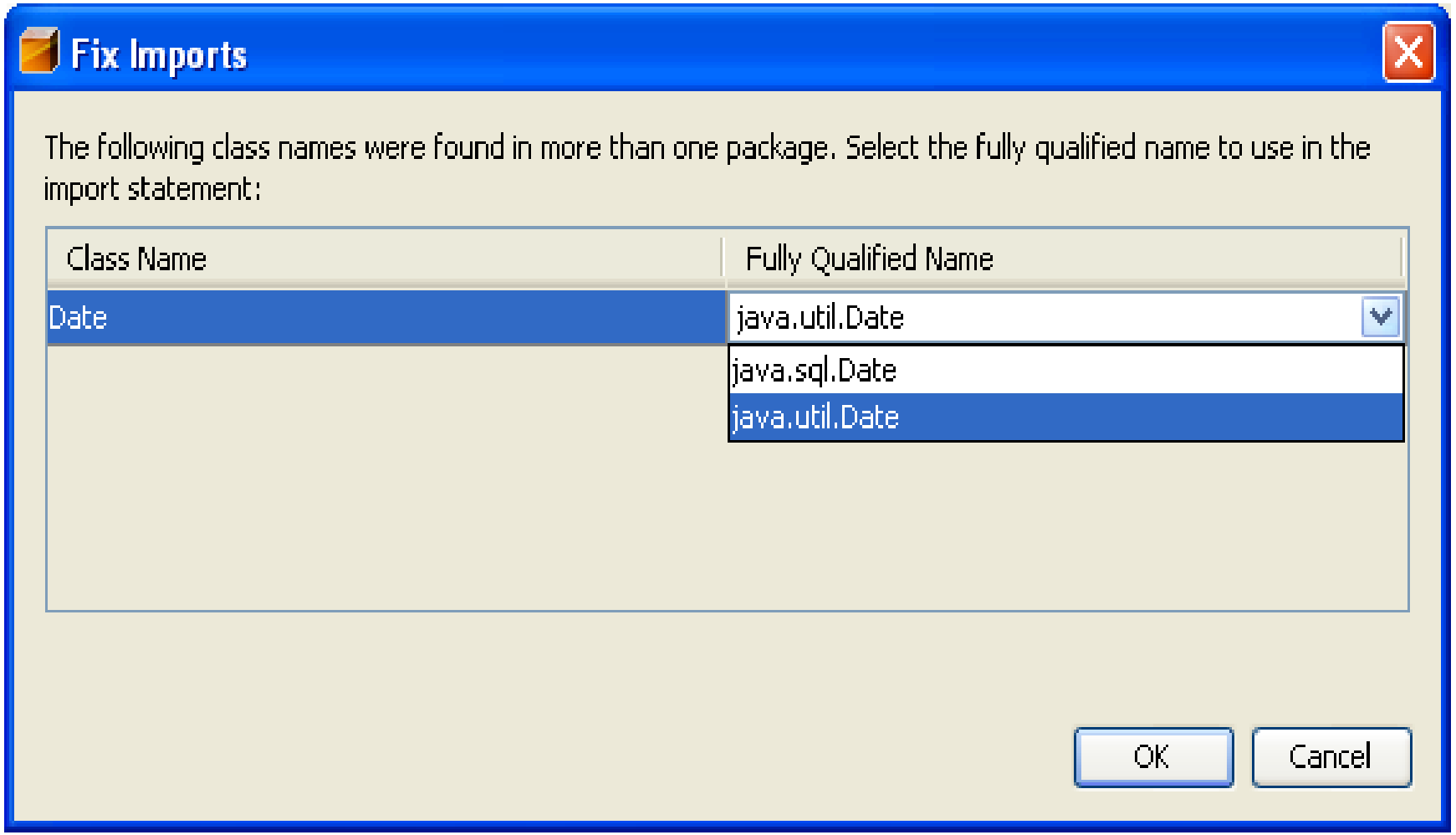
3.4. Uzupełnienie kodu metody init w klasie SessionBean1 – metoda init w każdym obiekcie uruchamiana jest w momencie tworzenia obiektu i przypisuje wartość false do atrybutu rejestracja



4. Dodawanie atrybutu czas typu Date do klasy typu RequestBean1 – ręczne wstawienie kodu



4.1. Import brakujących pakietów - klawisze CTRL+Shift+I oraz wybór w formularzu, który pokazuje się w sytuacji, kiedy należy wybrać właściwą klasę

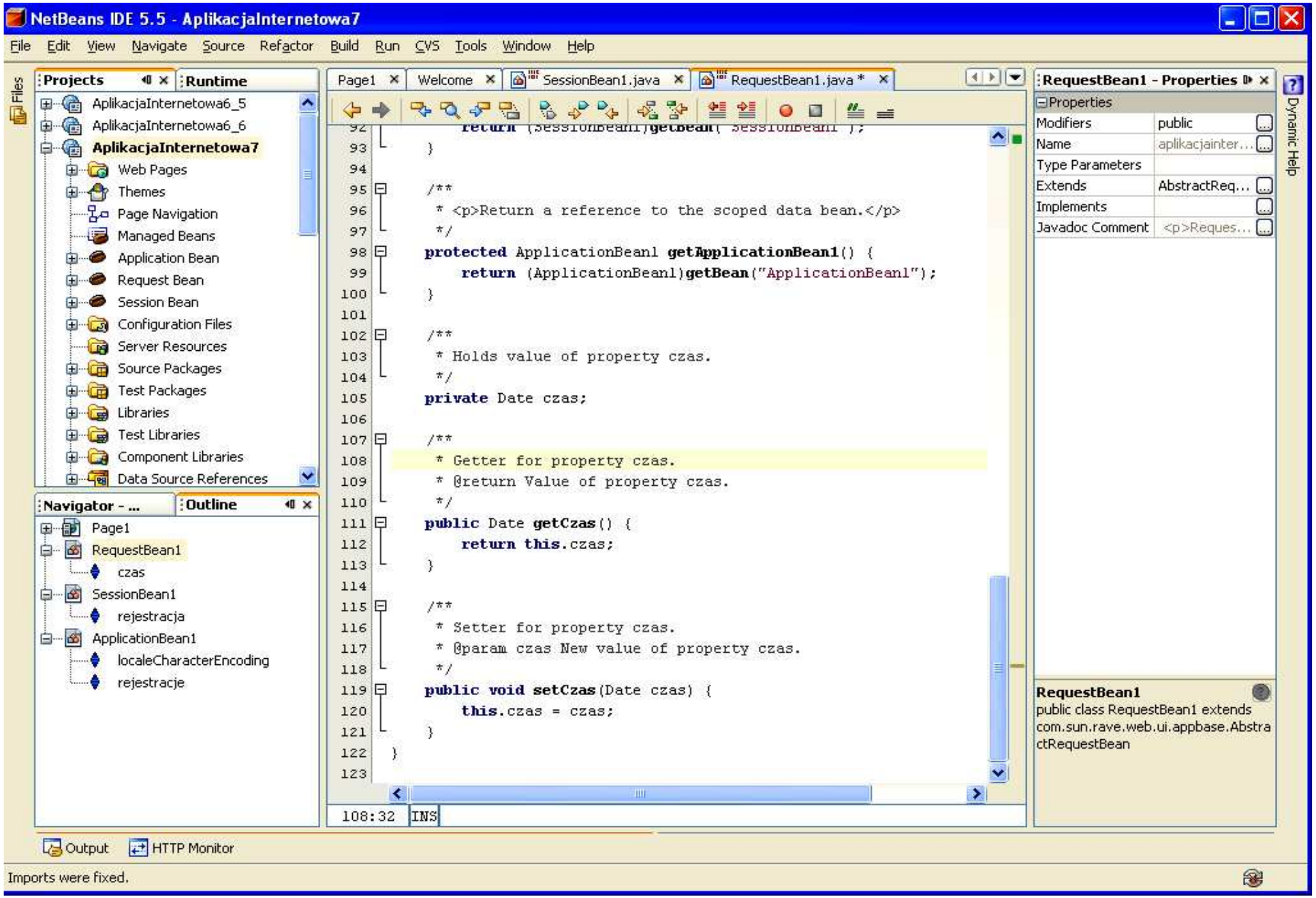


4.2. Dodanie atrybutu czas typu Date do klasy RequestBean1 w celu przechowania czasu rejestracji klienta pamiętany jedynie podczas przesłania jego wartości oraz danych rejestracji ze strony głównej Page1 do strony Rejestracje. Wartość atrybutu czas zawiera odczytany czas systemowy – kliknąć prawym klawiszem myszy na ciało klasy w trybie Java, wybrać Refactor, następnie Encapsulate Fields.

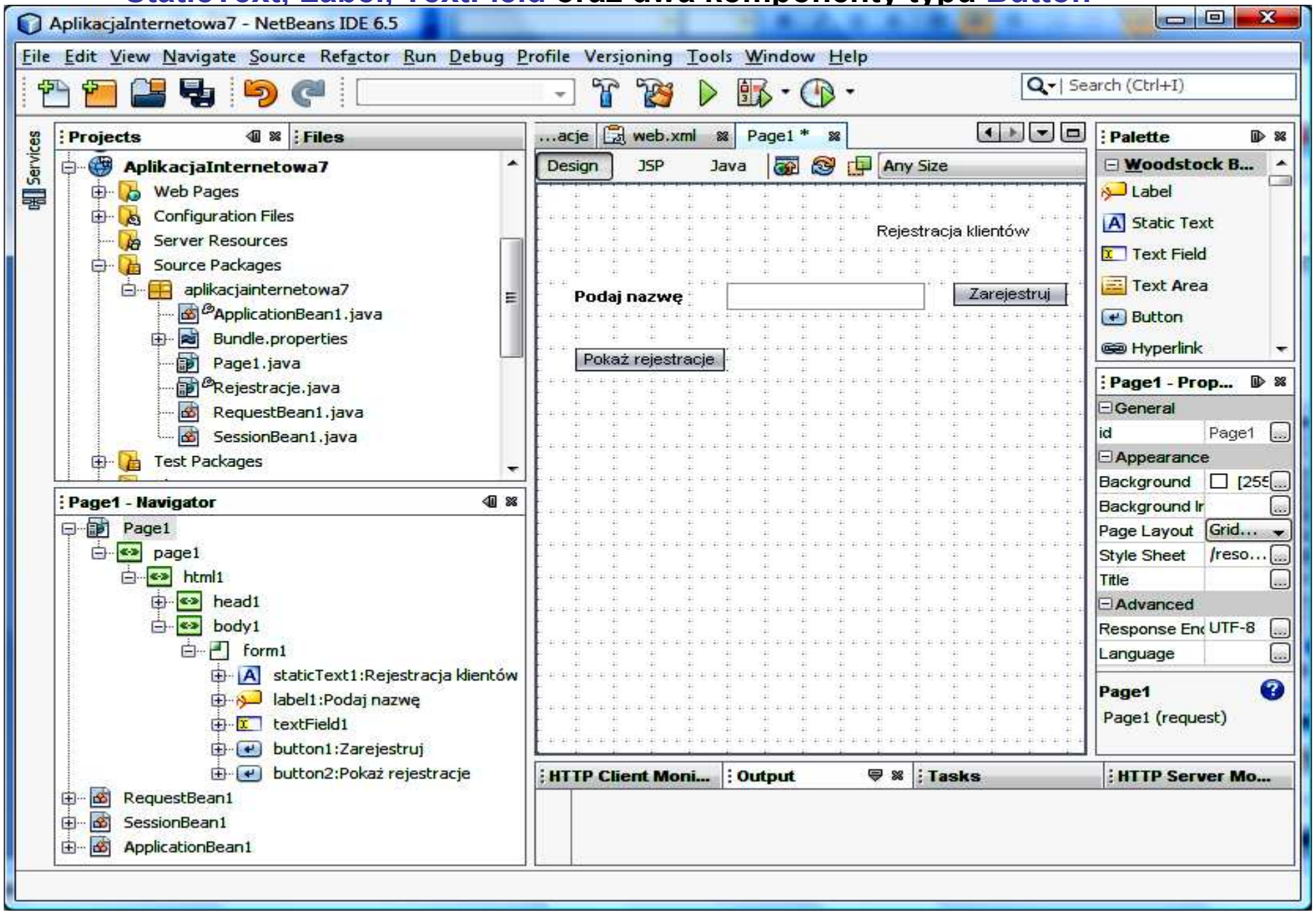
4.3. Dodanie metod typu set i get dla atrybutu czas typu Date do klasy RequestBean1 w celu przechowania czasu rejestracji klienta pamiętany jedynie podczas przesłania jego wartości oraz danych rejestracji ze strony głównej Page1 do strony Rejestracje cd. – zaznaczyć Create Getter oraz Create Setter i nacisnąć **Refactor**

Uwagi
W momencie wywołania strony Page1 tworzony jest obiekt typu RequestBean1. Odczytany wtedy czas systemowy zapisany w atrybucie czas typu Date jest pamiętany do momentu, kiedy kończy się obsługa wysłania wartości tego czasu na stronę Rejestracje.

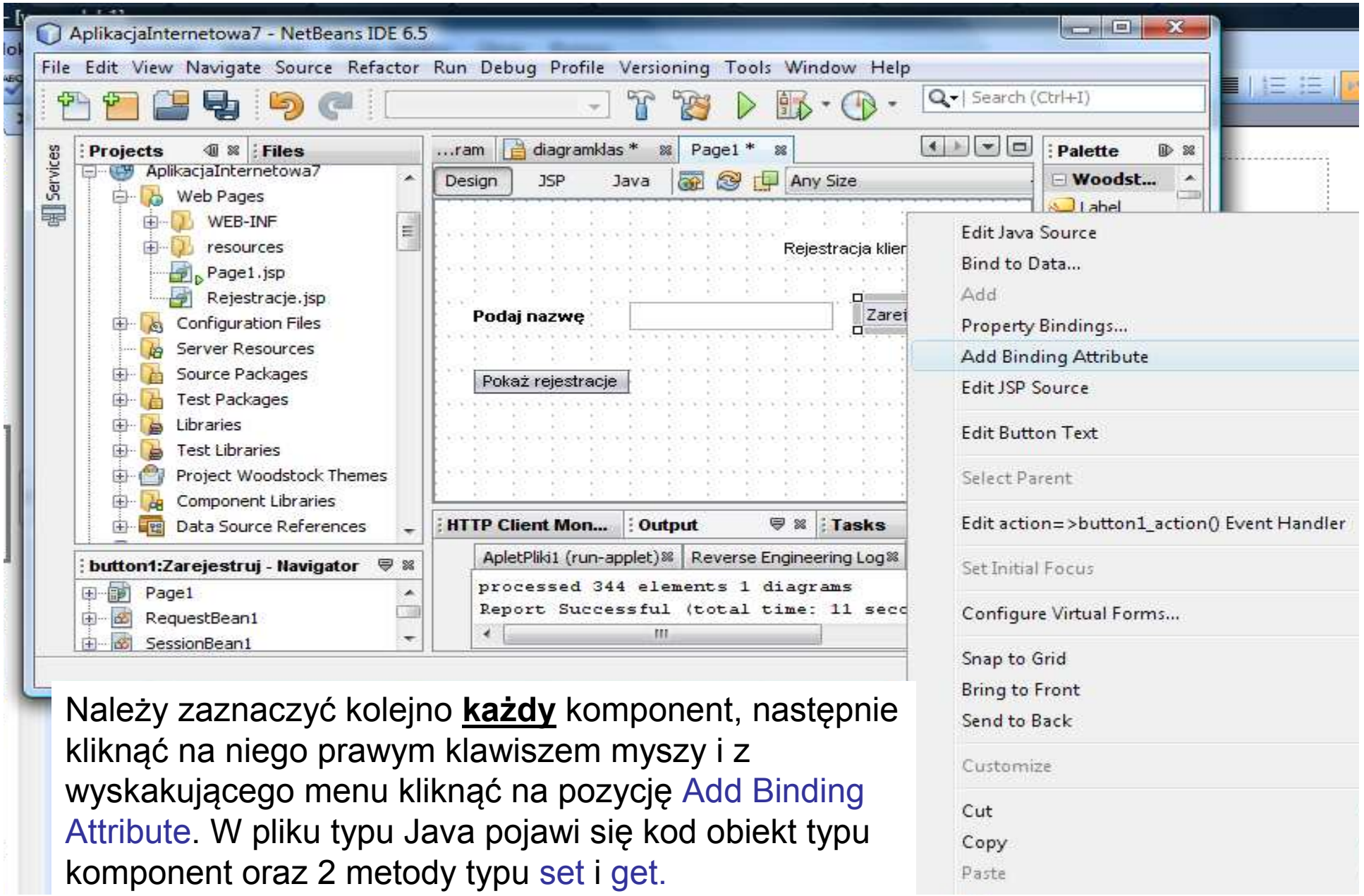
4.4. Rezultat działań – obiekt RequestBean1 posiada atrybut czas typu Date



5. Wstawienie komponentów w trybie Design do strony Page1 typu: StaticText, Label, TextField oraz dwa komponenty typu Button

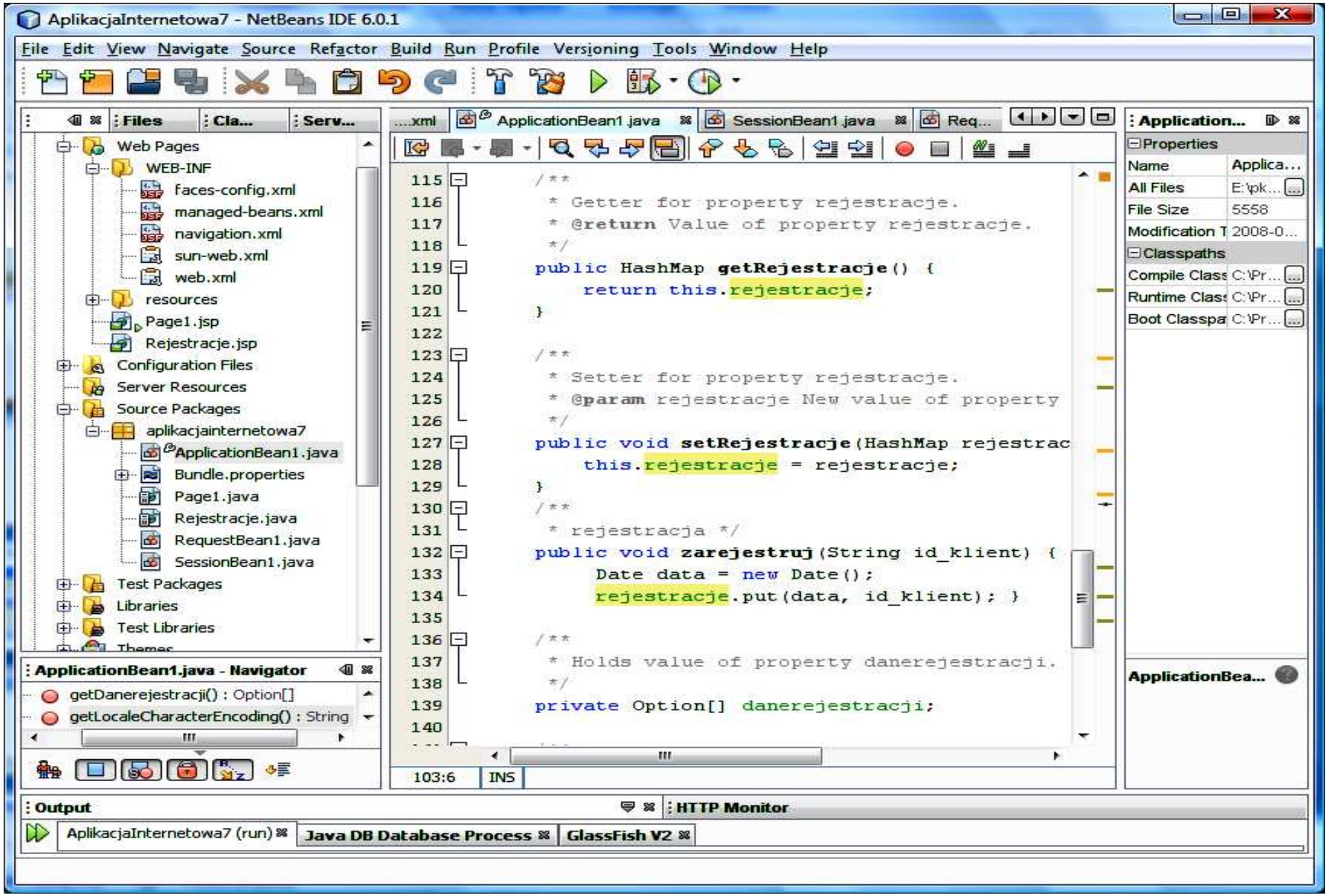


5.1. Wstawianie do pliku typu Java strony JSP obiektów reprezentujących komponenty wizualne interfejsu graficznego użytkownika

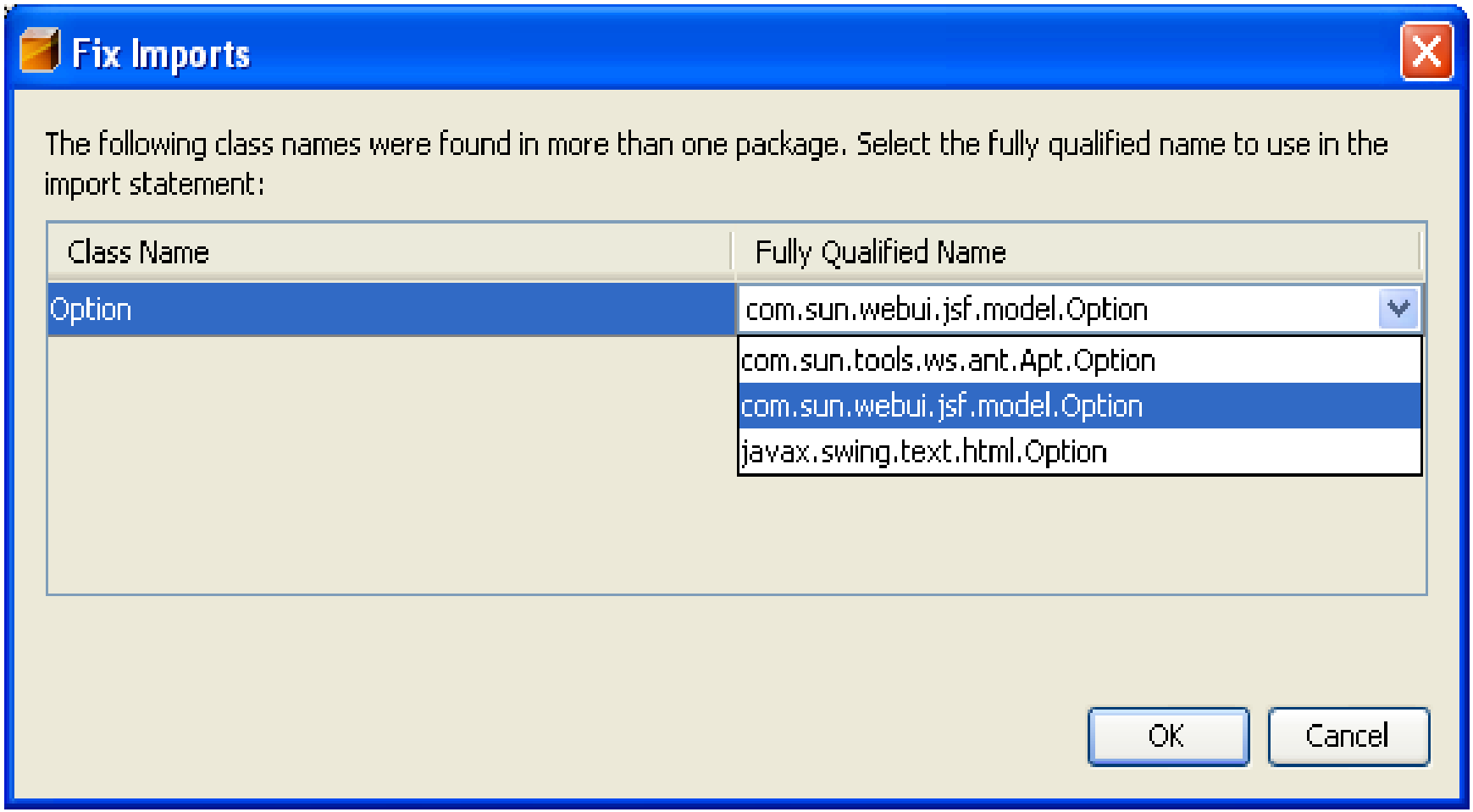


Należy zaznaczyć kolejno **<u>każdy</u>** komponent, następnie kliknąć na niego prawym klawiszem myszy i z wyskakującego menu kliknąć na pozycję Add Binding Attribute. W pliku typu Java pojawi się kod obiekt typu komponent oraz 2 metody typu set i get.

6. Przygotowanie do wstawienia atrybutu danerejestracji jako tablicy elementów Option do klasy typu ApplicationBean1 – ręczne wstawienie atrybutu



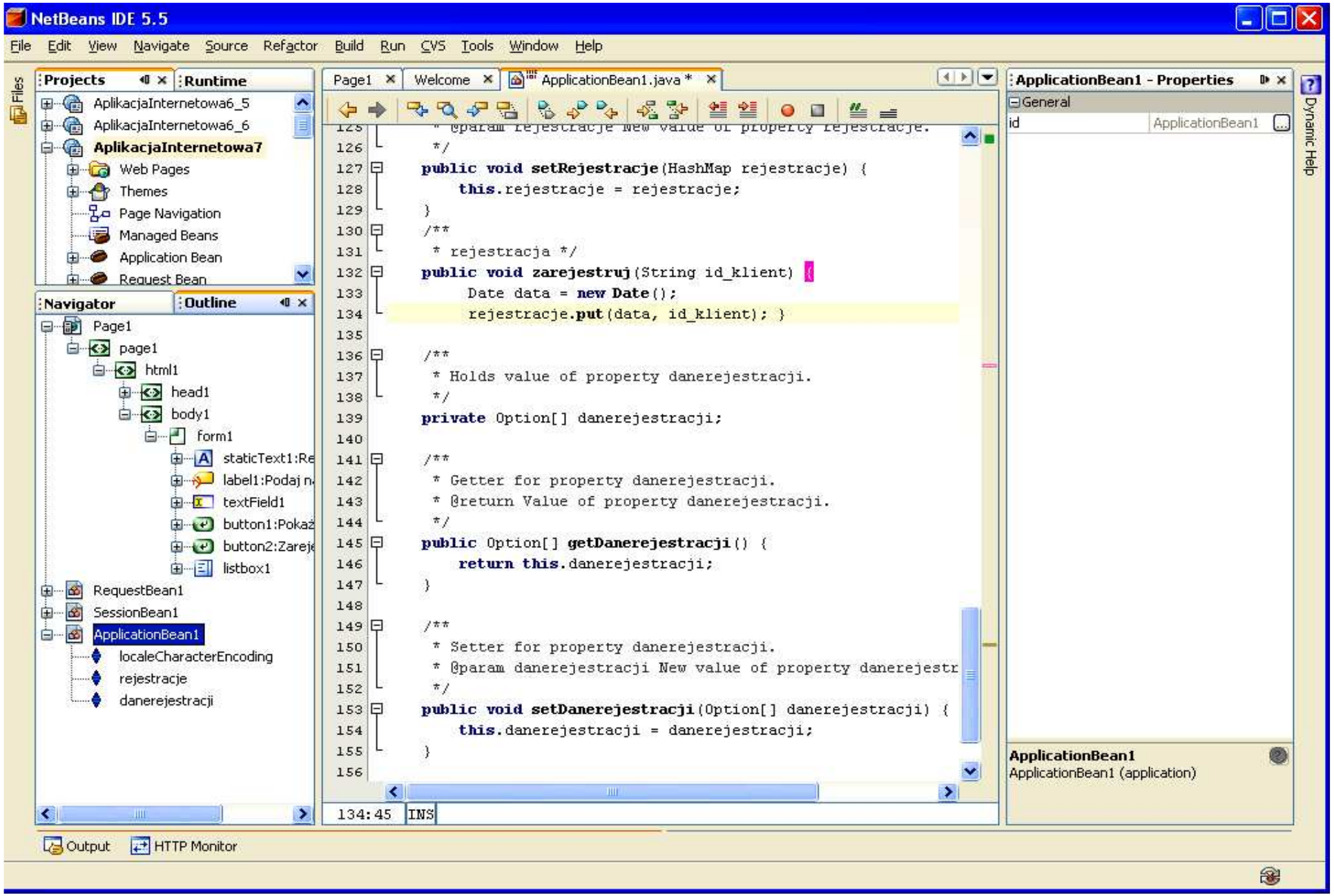
6.1. Import brakujących pakietów - klawisze CTRL+Shift+I oraz wybór w formularzu, który pokazuje się w sytuacji, kiedy należy wybrać właściwą klasę



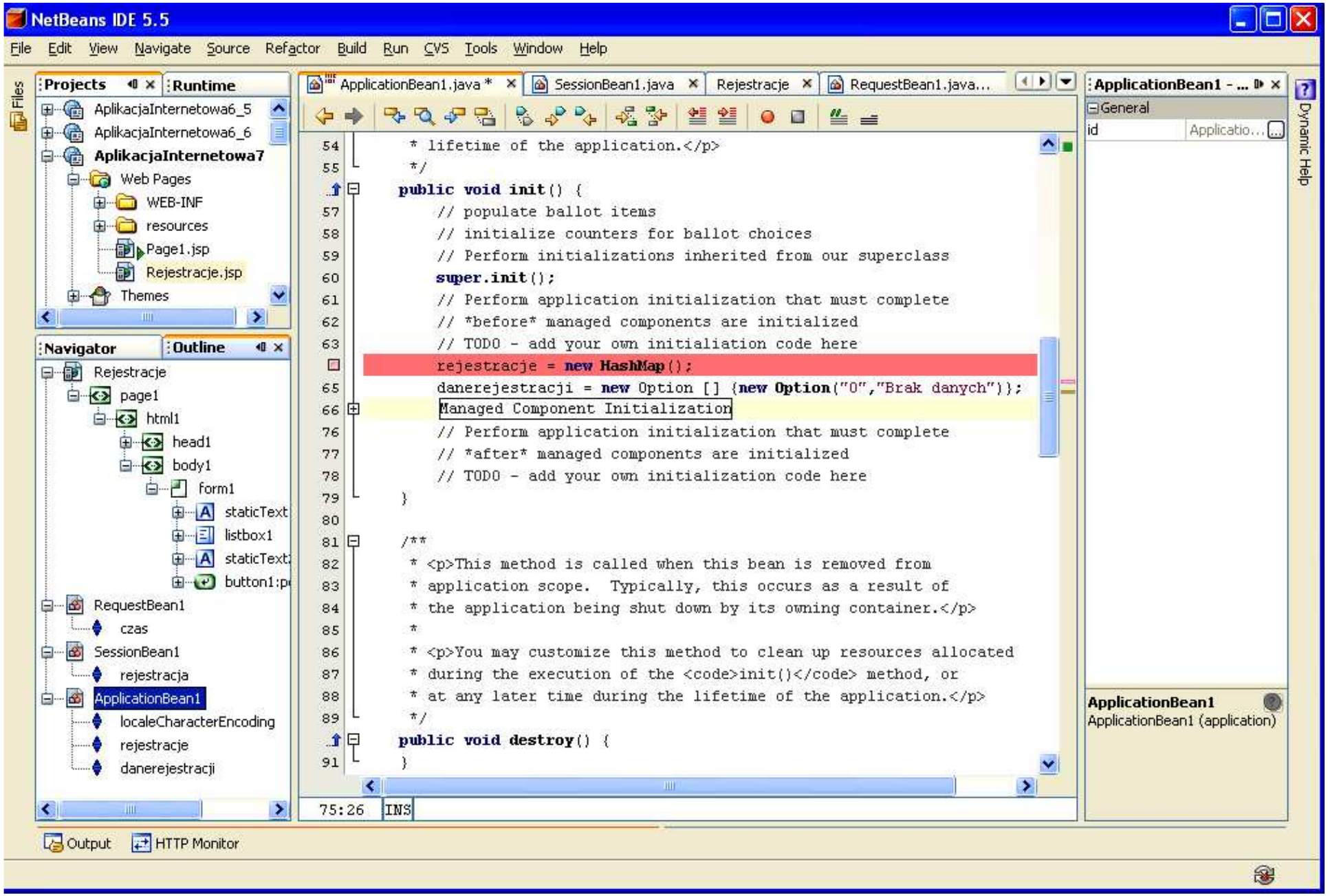
6.2. Wstawienie atrybutu danerejestracji jako tablicy elementów Option do klasy typu ApplicationBean1 w celu umożliwienia wyświetlania danych rejestracji przechowywanych w obiekcie ApplicationBean1 w atrybucie rejestracje typu HasMap – kliknąć prawym klawiszem myszy na ciało klasy w trybie Java, wybrać Refactor, następnie Encapsulate Fields.

6.3. Dodanie metod typu set i get dla atrybutu danerejestracji, jako tablicy elementów Option, do klasy typu ApplicationBean1 w celu umożliwienia wyświetlania danych rejestracji przechowywanych w obiekcie ApplicationBean1 w atrybucie rejestracje typu HasMap cd. – zaznaczyć Create Getter oraz Create Setter i nacisnąć **Refactor**

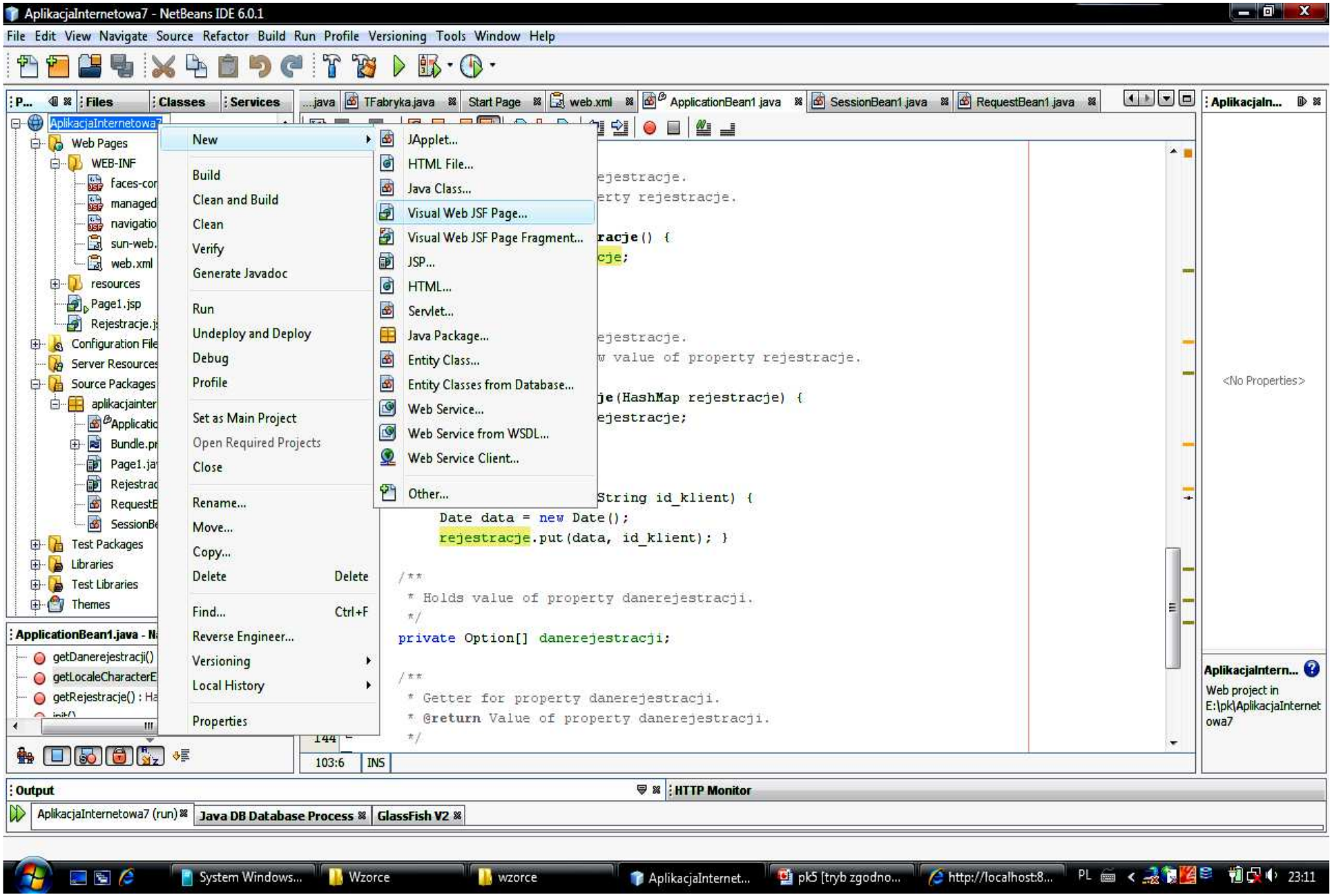
6.4. Rezultat po wstawieniu atrybutu danerejestracji jako tablicy elementów Option do klasy typu ApplicationBean1



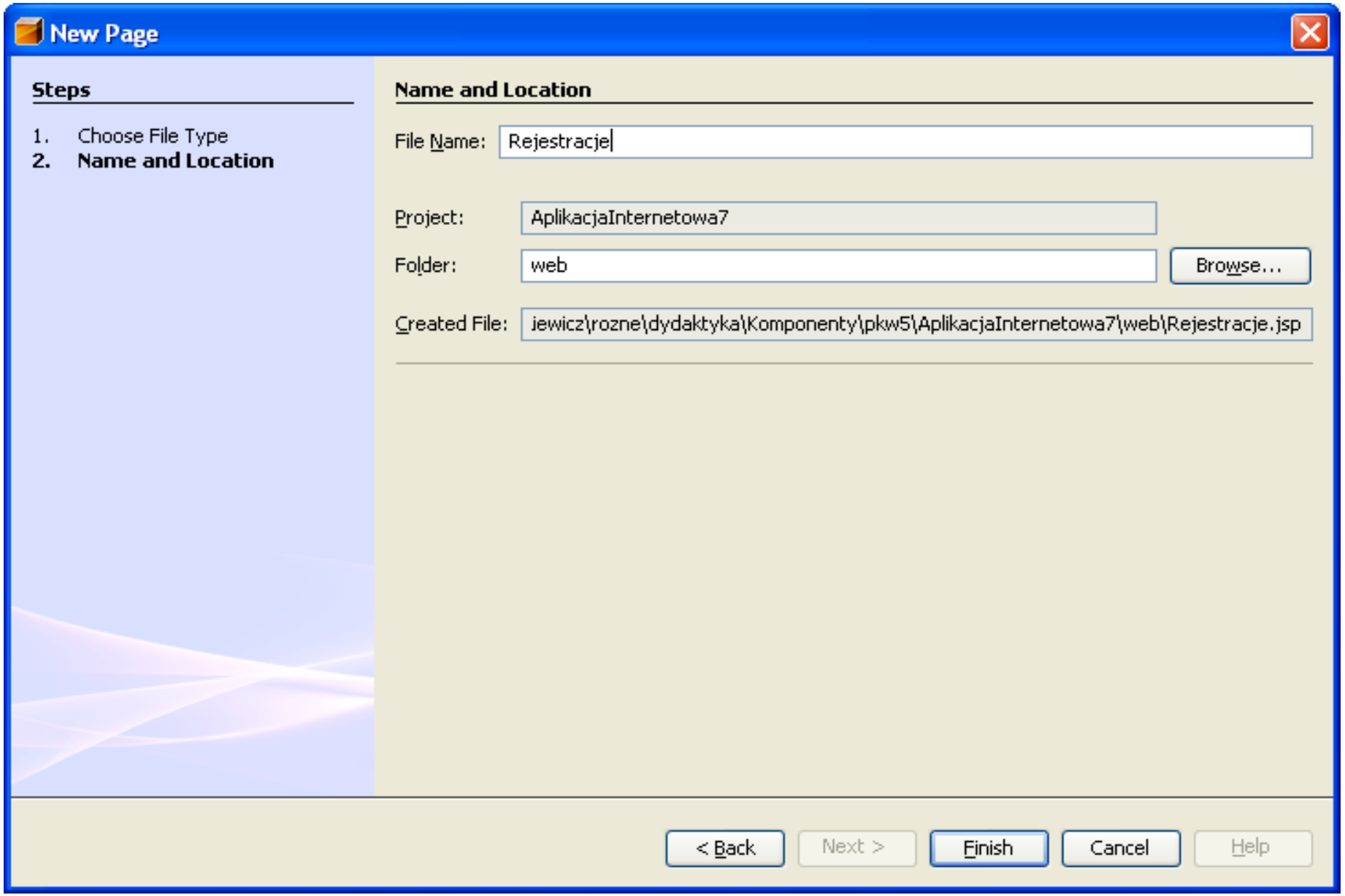
6.5. Uzupełnienie metody init w klasie typu ApplicationBean1 – inicjowanie tablicy danerejestracji



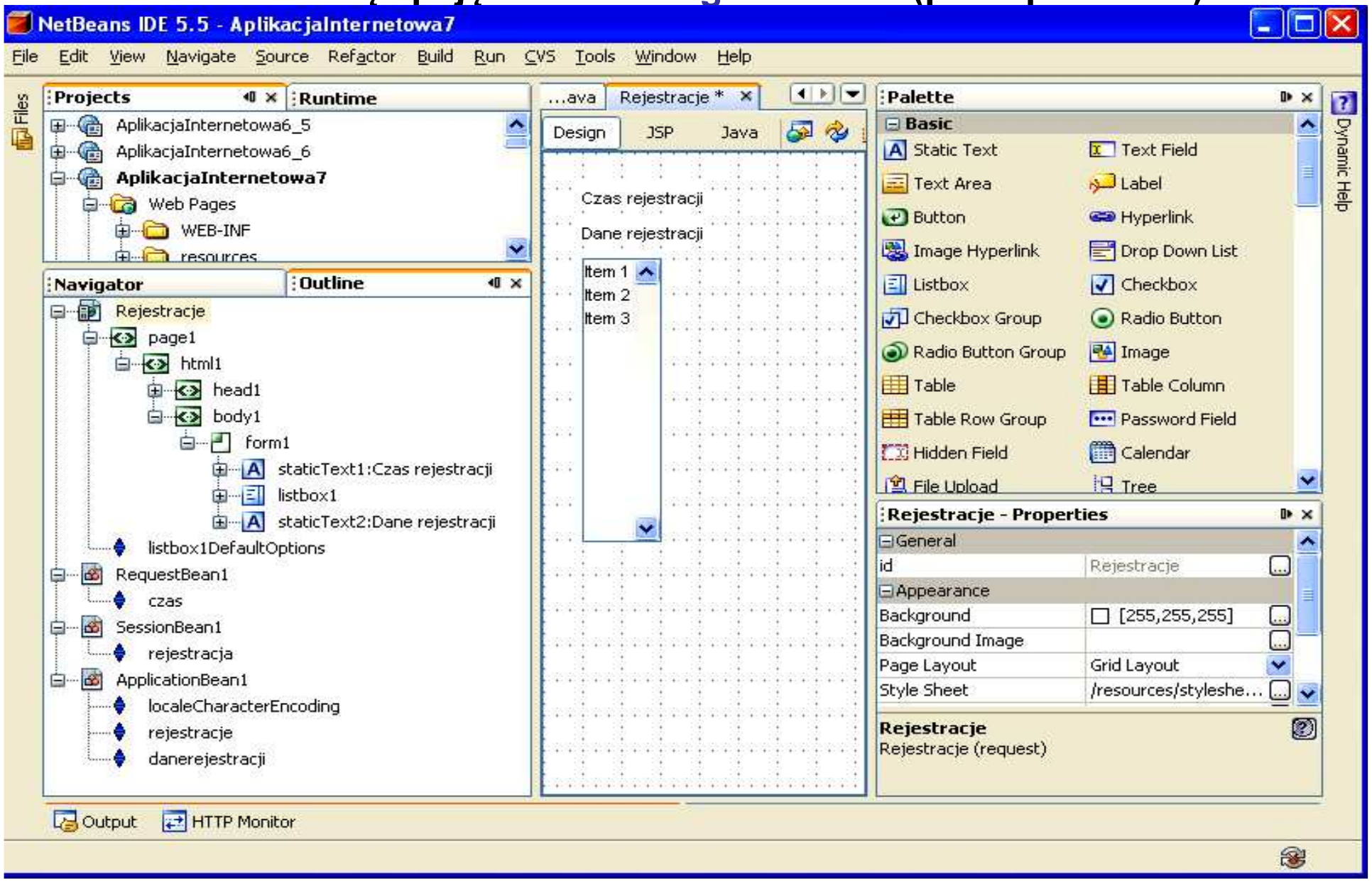
7. Dodanie nowej strony Rejestracje.jsp do prezentowania danych aplikacji oraz czasu życia atrybutów obiektów aplikacji



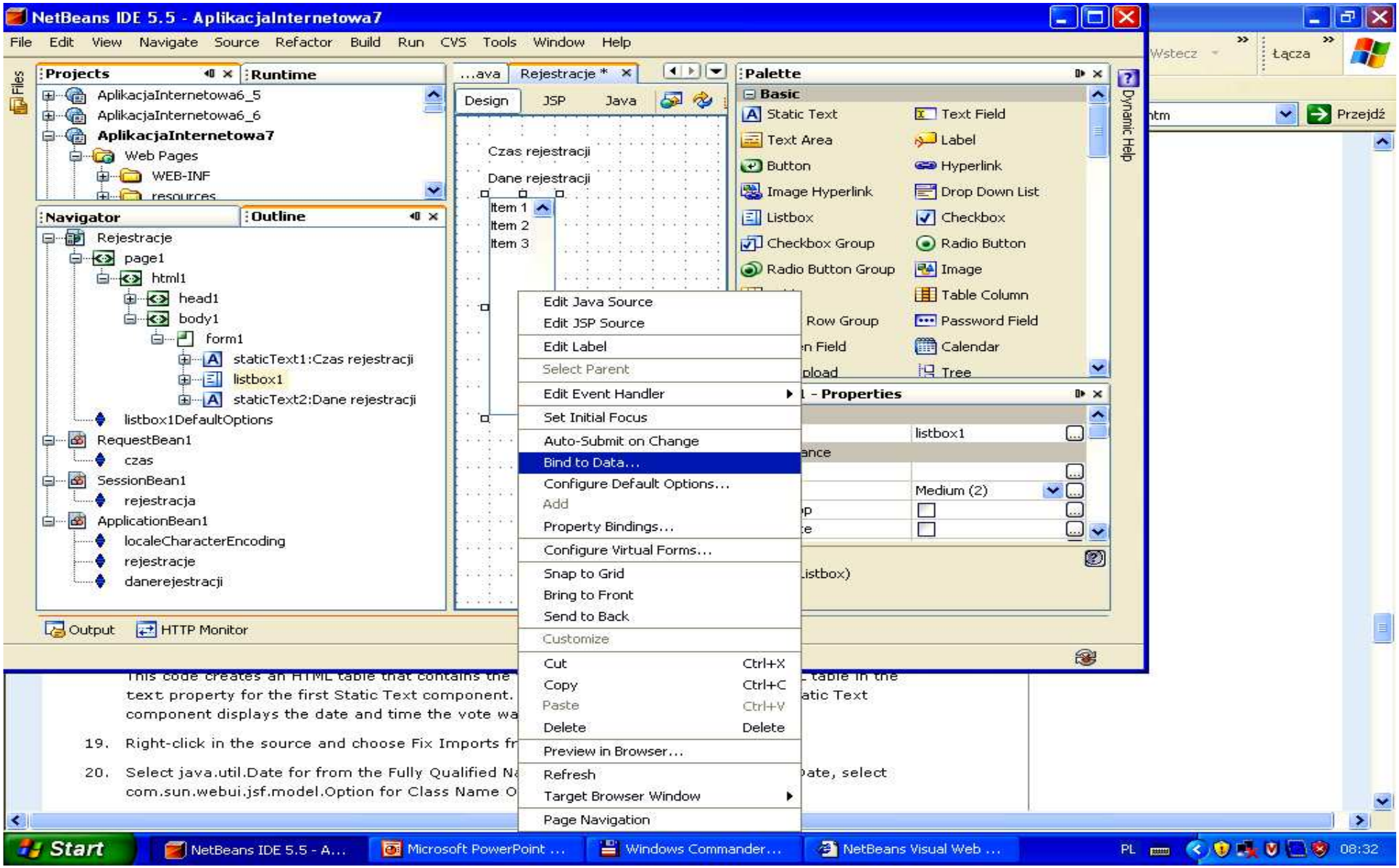
7.1. Dodanie nowej strony Rejestracje.jsp do prezentowania danych aplikacji oraz wywołania strony



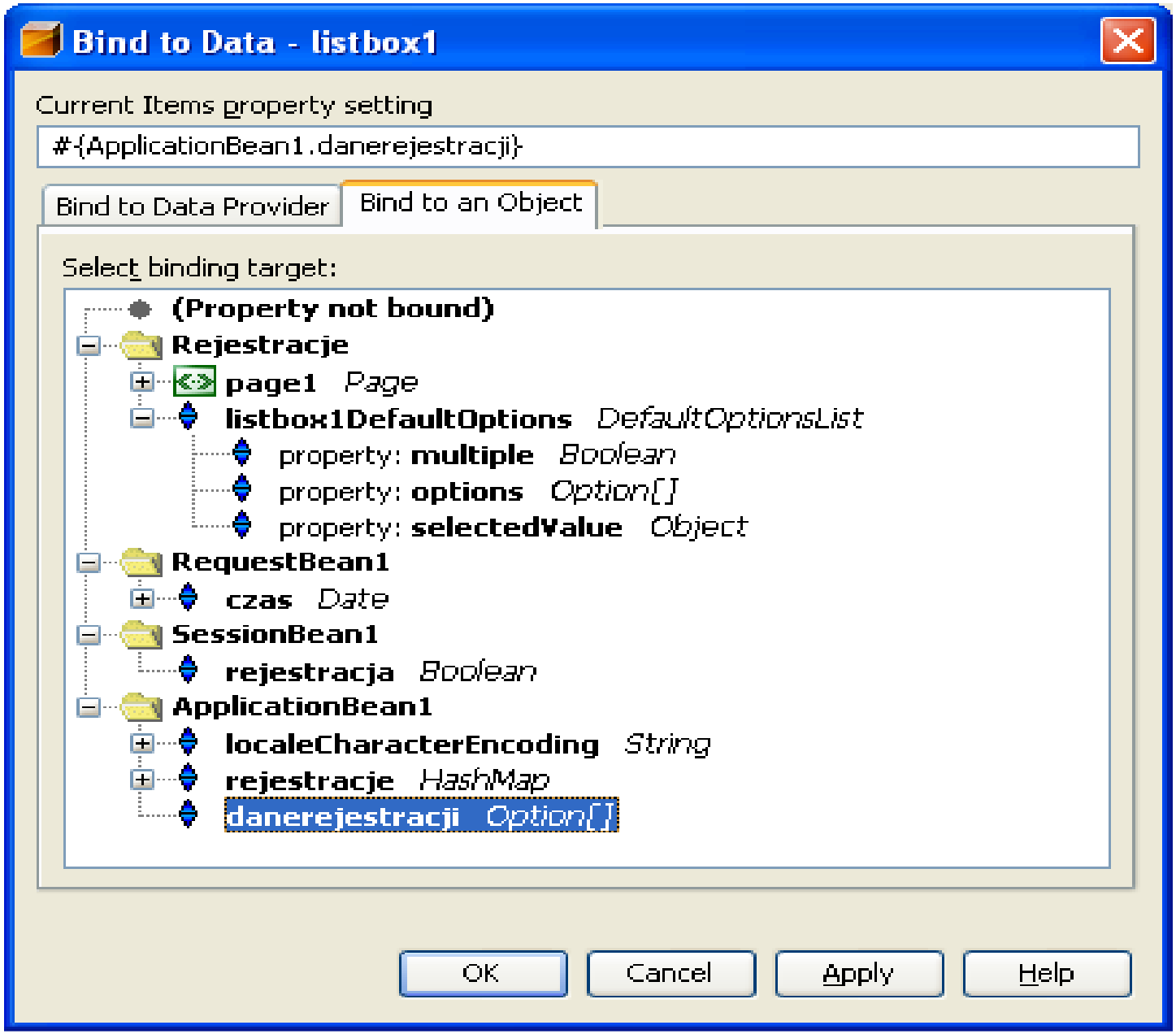
8. Wstawienie komponentów do strony Rejestracje – dwa typu StaticText oraz Listbox. Każdy komponent powinien mieć ustawioną opcję Add Binding Attribute (patrz punkt 5.1)



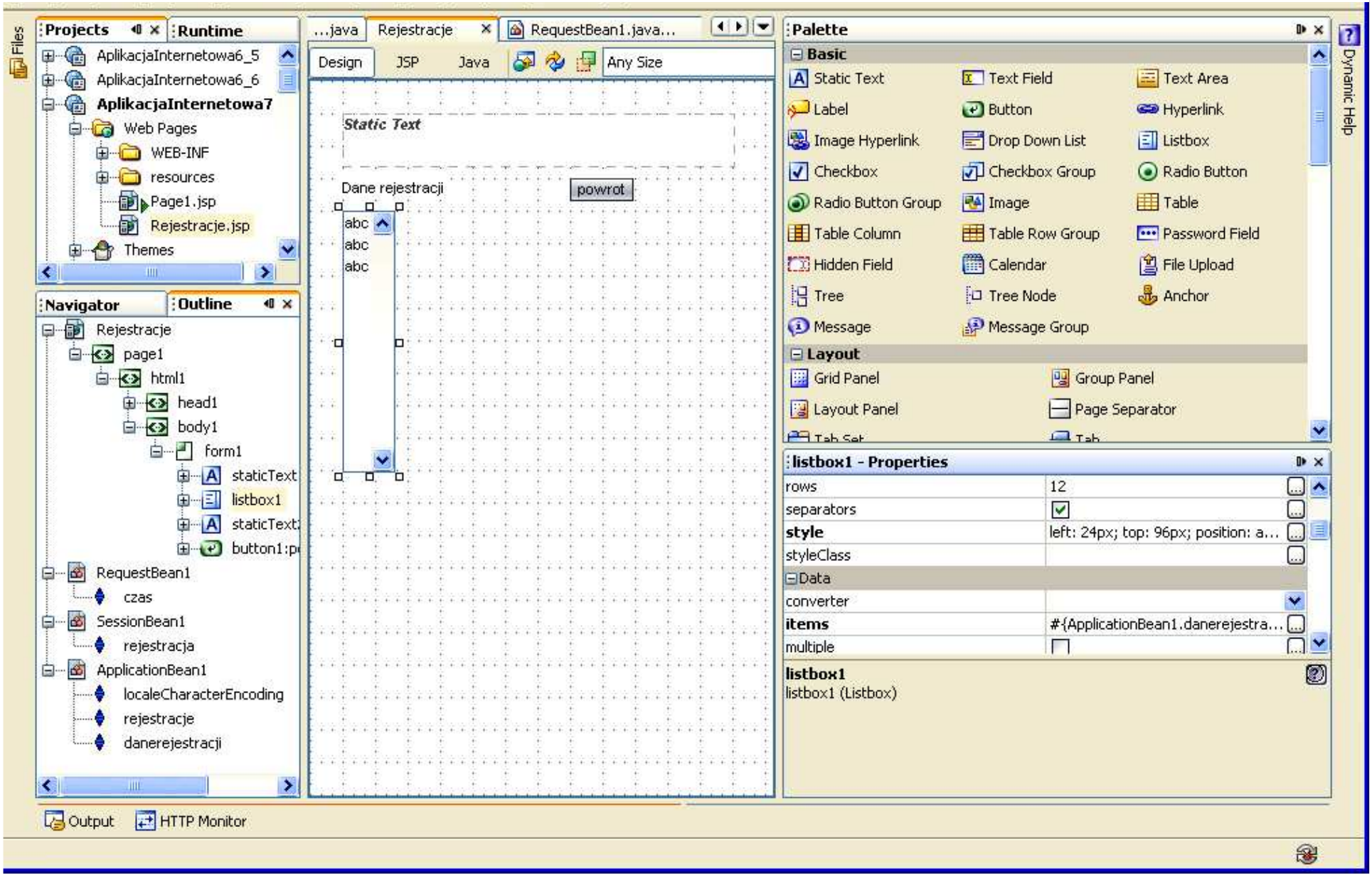
8.1. Tworzenie połączenia komponentu typu List Box z obiektem danerejestracji z obiektu typu ApplicationBean1 – wywołanie opcji Bind to Data dla komponentu typu ListBox z wyskakującego menu



8.2. Połączenie komponentu typu ListBox z obiektem danerejestracji z obiektu typu ApplicationBean1



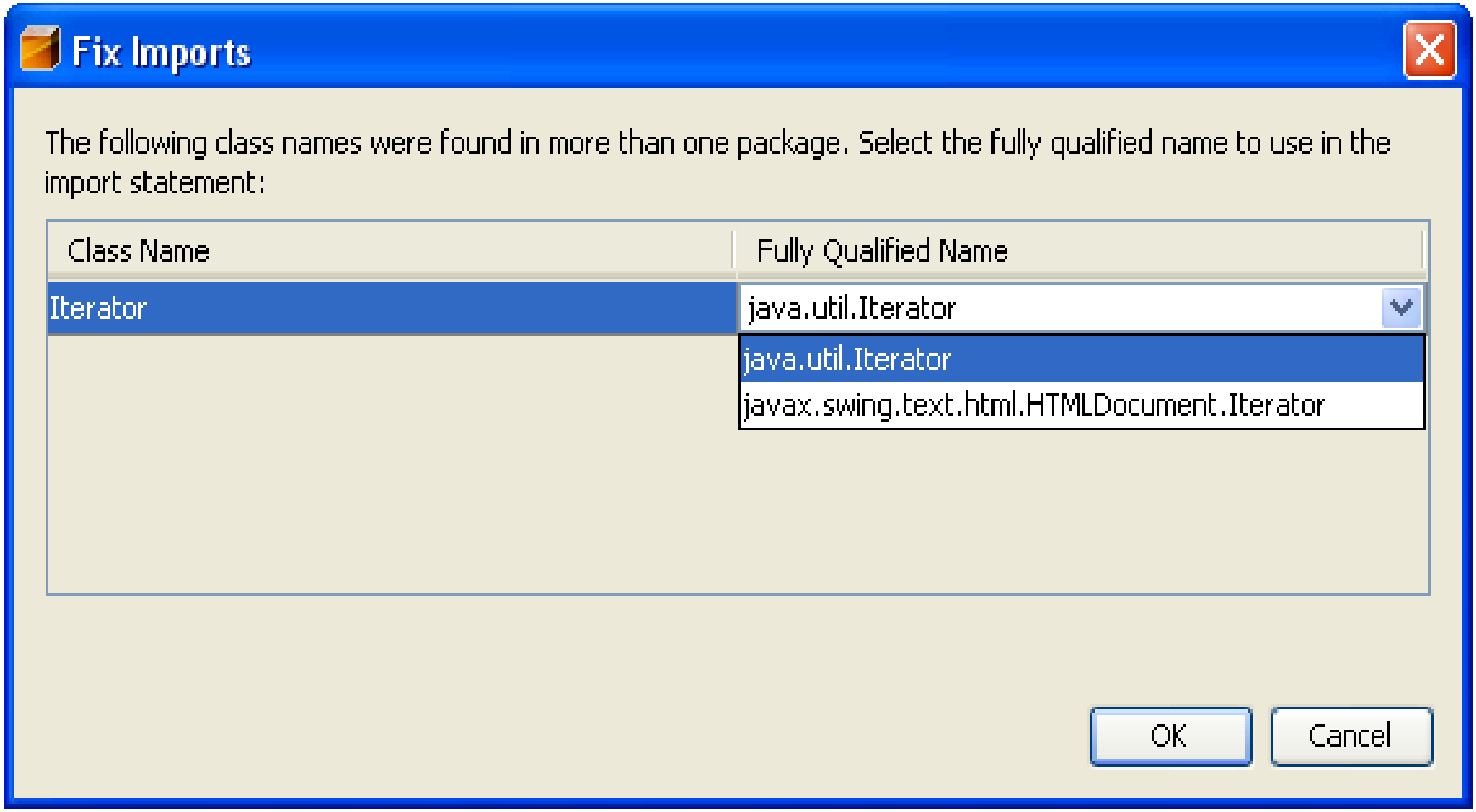
8.3. Wstawienie przycisku powrot typu Button do strony Rejestracje. Powinien on mieć ustawioną opcję Add Binding Attribute (patrz punkt 5.1)



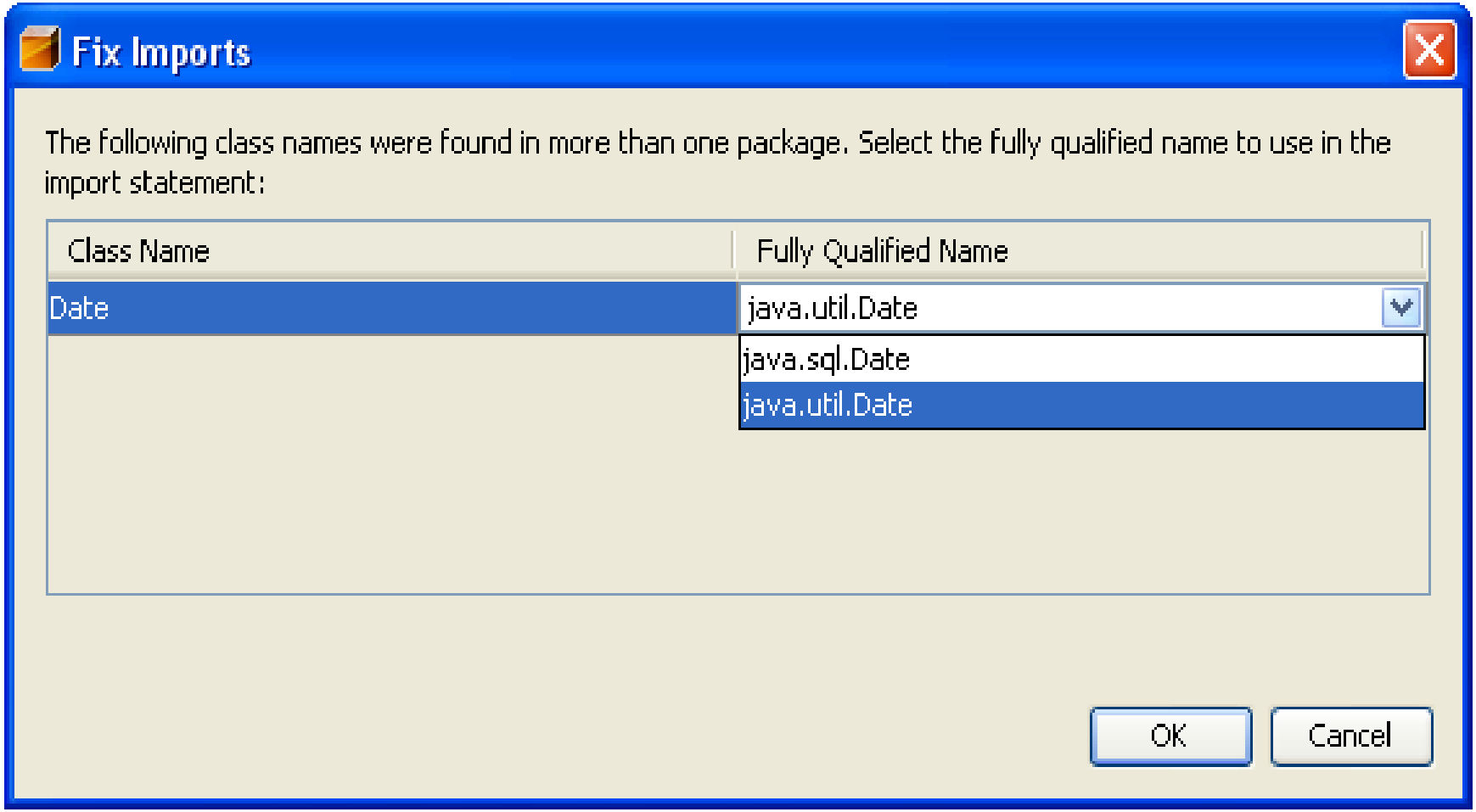
8.4. Wstawienie metody zawartosc_tablicy_danerejestracji() do klasy ApplicationBean1 przygotowującej tablicę danerejestracji do „bindowania” z komponentem typu ListBox ze strony Rejestracje

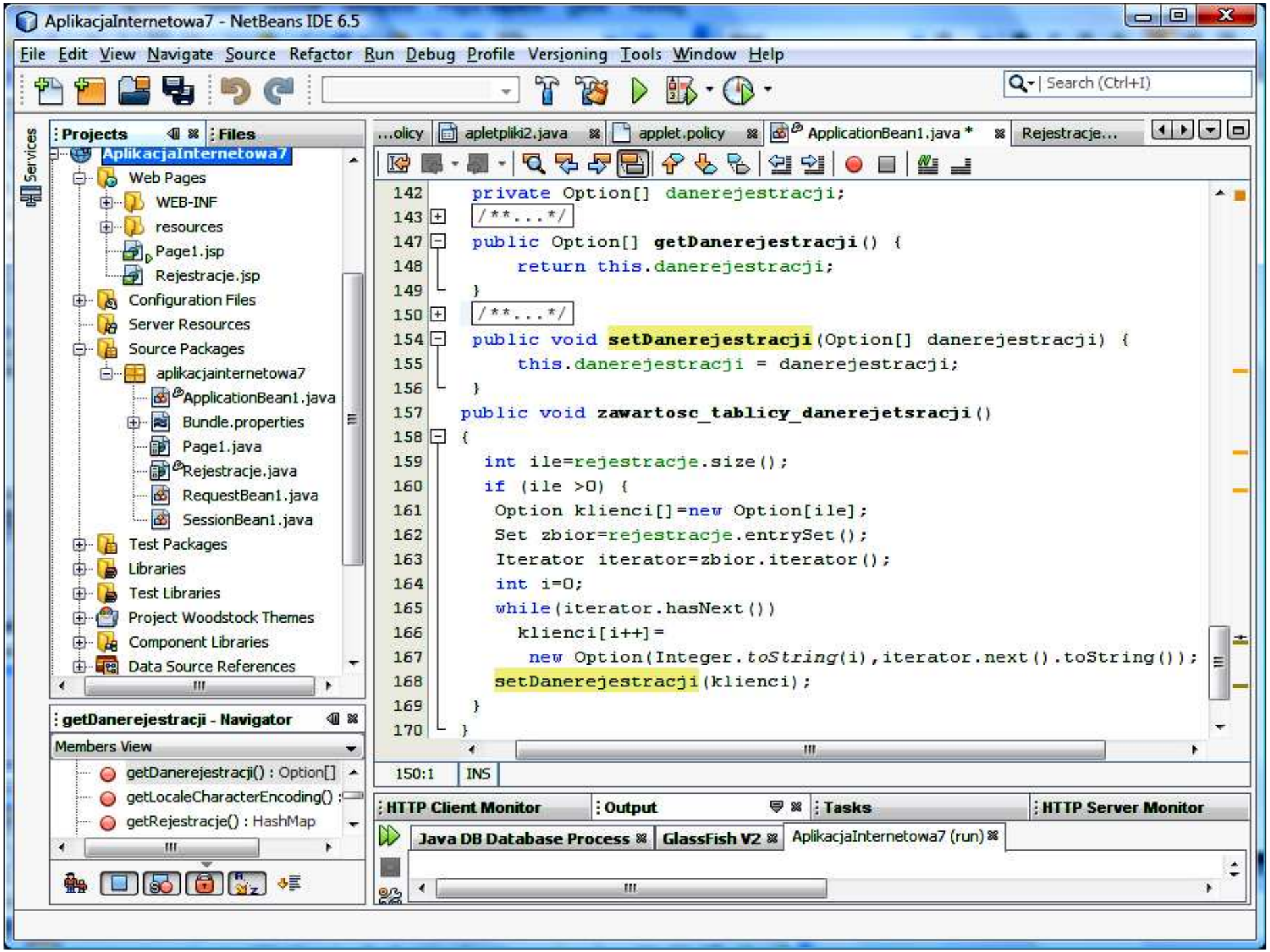
```
public void zawartosc_tablicy_danerejestracji()
  {
    int ile = rejestracje.size();
    if (ile >0) {
     Option klienci[] = new Option[ile];
     Set zbior = rejestracje.entrySet();
     Iterator iterator = zbior.iterator();
     int i=0;
     while(iterator.hasNext())
       klienci[i++] =
           new Option(Integer.toString(i),iterator.next().toString());
     setDanerejestracji(klienci);
    }
 }
```

8.5. Import brakujących pakietów - klawisze CTRL+Shift+I oraz wybór w formularzu, który pokazuje się w sytuacji, kiedy należy wybrać właściwą klasę



8.6. Import brakujących pakietów - klawisze CTRL+Shift+I oraz wybór w formularzu, który pokazuje się w sytuacji, kiedy należy wybrać właściwą klasę

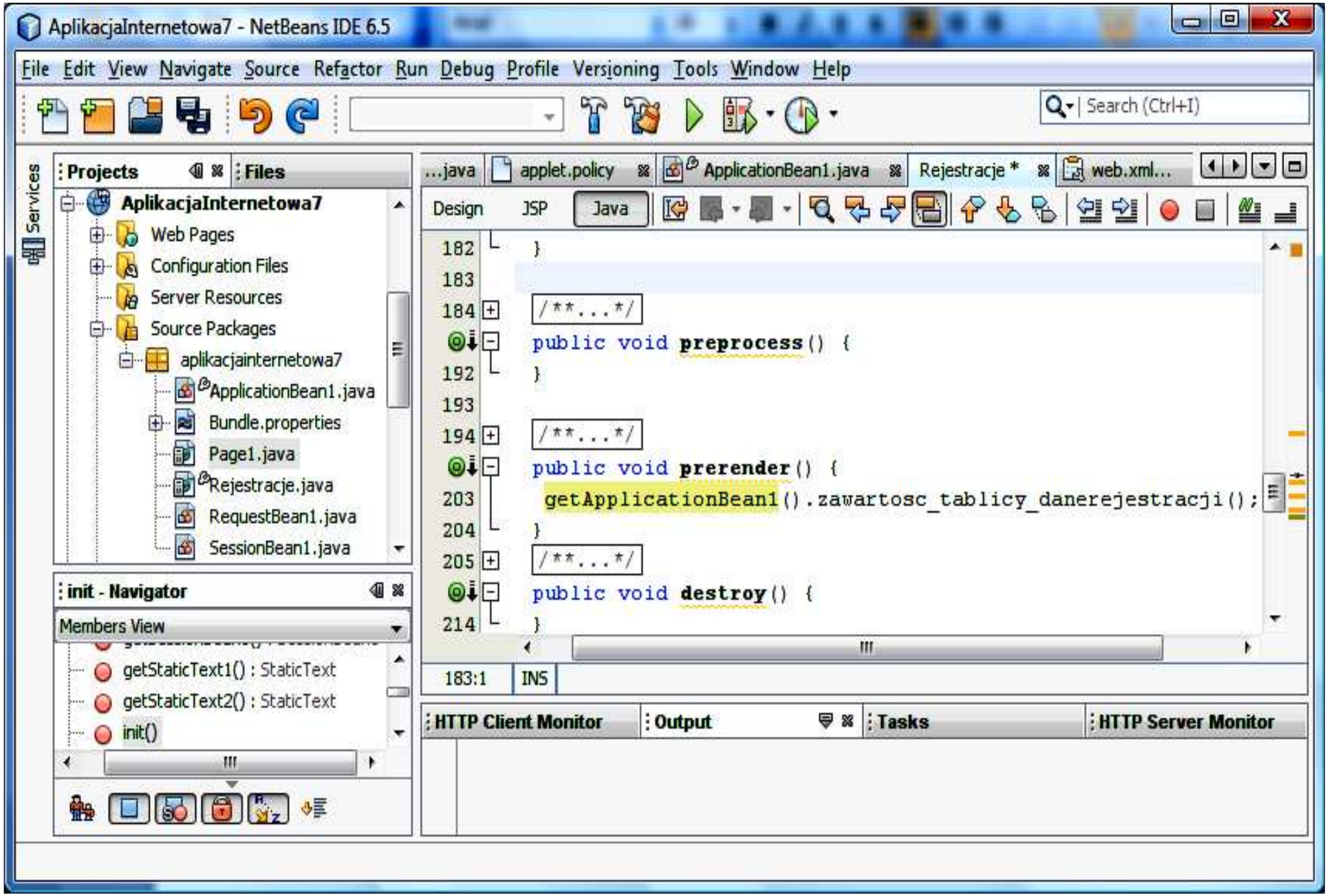



AplikacjaInternetowa7 - NetBeans IDE 6.5
File Edit View Navigate Source Refactor Run Debug Profile Versioning Tools Window Help
Search (Ctrl+I)
Projects
Files
Services
AplikacjaInternetowa7
Web Pages
WEB-INF
resources
Page1.jsp
Rejestracje.jsp
Configuration Files
Server Resources
Source Packages
aplikacjainternetowa7
ApplicationBean1.java
Bundle.properties
Page1.java
Rejestracje.java
RequestBean1.java
SessionBean1.java
Test Packages
Libraries
Test Libraries
Project Woodstock Themes
Component Libraries
Data Source References
getDanerejestracji - Navigator
Members View
getDanerejestracji() : Option[]
getLocaleCharacterEncoding() :
getRejestracje() : HashMap
...olicy
apletpliki2.java
applet.policy
ApplicationBean1.java *
Rejestracje...
142 private Option[] danerejestracji;
143 /**...*/
147 public Option[] getDanerejestracji() {
148 return this.danerejestracji;
149 }
150 /**...*/
154 public void setDanerejestracji(Option[] danerejestracji) {
155 this.danerejestracji = danerejestracji;
156 }
157 public void zawartosc_tablicy_danerejetsracji()
158 {
159 int ile=rejestracje.size();
160 if (ile >0) {
161 Option klienci[]=new Option[ile];
162 Set zbior=rejestracje.entrySet();
163 Iterator iterator=zbior.iterator();
164 int i=0;
165 while(iterator.hasNext())
166 klienci[i++]=
167 new Option(Integer.toString(i),iterator.next().toString());
168 setDanerejestracji(klienci);
169 }
170 }
150:1
INS
HTTP Client Monitor
Output
Tasks
HTTP Server Monitor
Java DB Database Process
GlassFish V2
AplikacjaInternetowa7 (run)

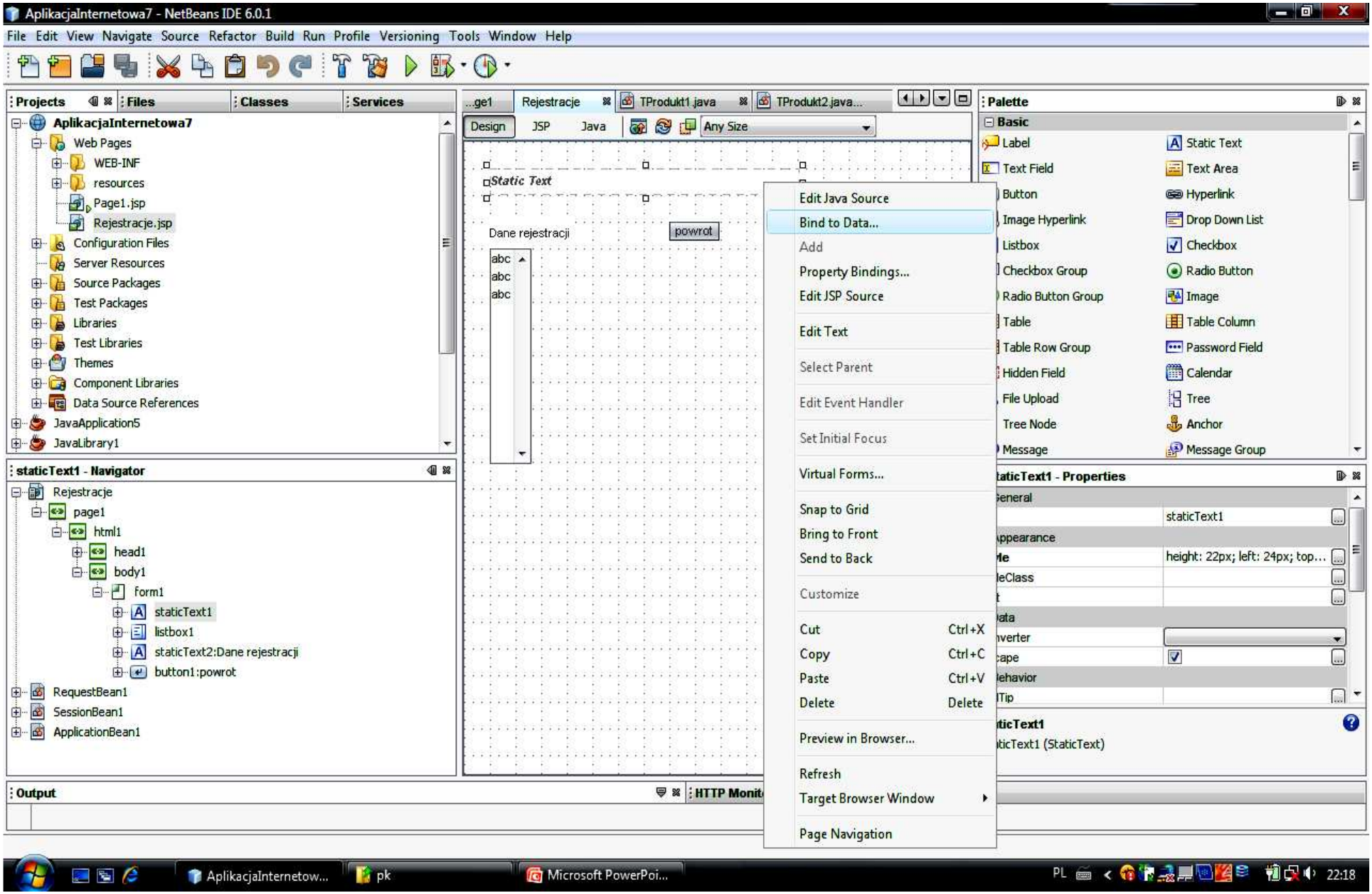
8.7. Uzupełnienie kodu metody istniejącej metody prerender dla strony Rejestracje (plik Rejestracja.java) – metoda wykonywana podczas wywoływania strony za pomocą przycisków Zarejestruj oraz Pokaż rejestracje strony głównej Page1

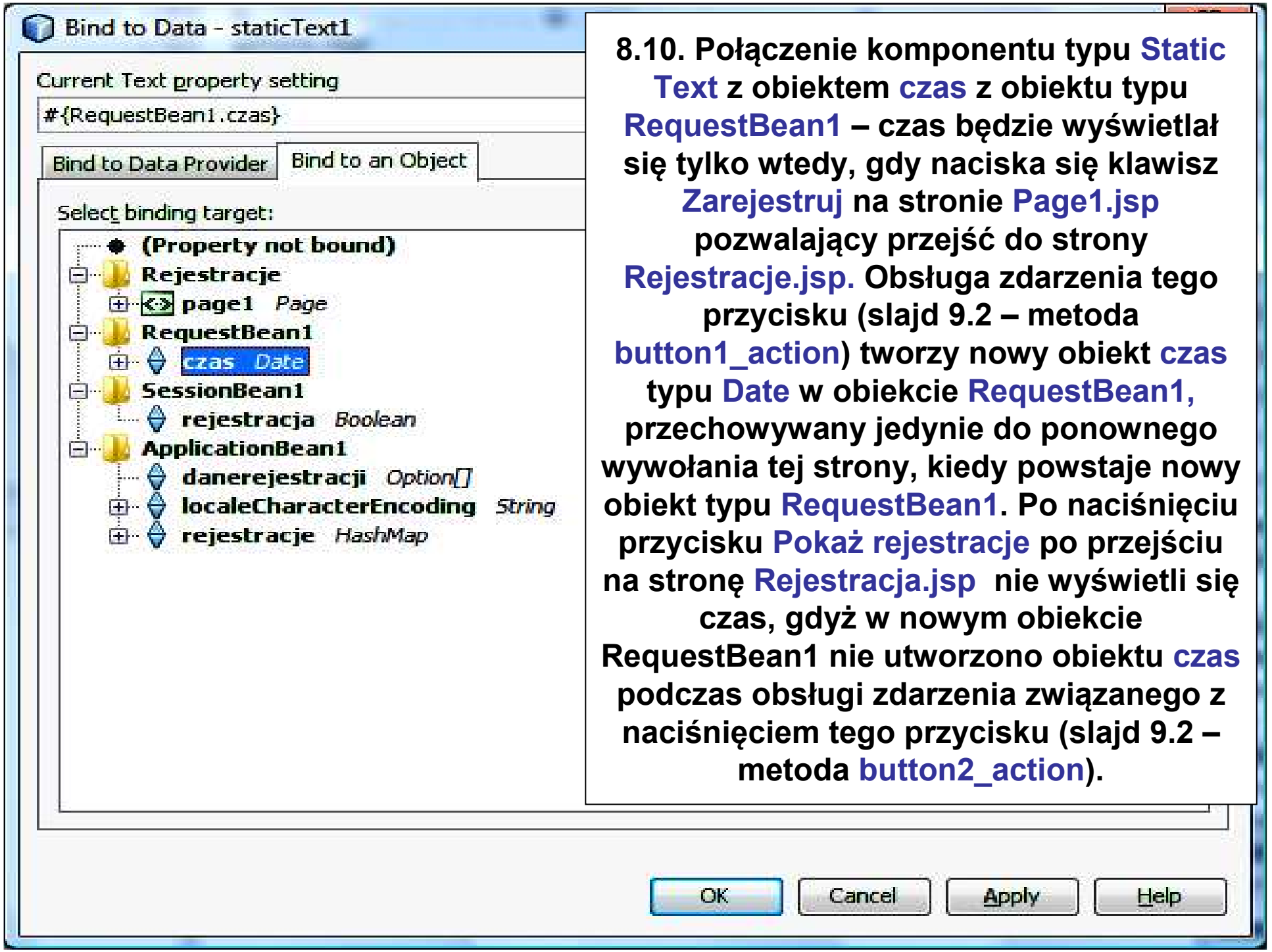
```
public void prerender()
 {
    getApplicationBean1().zawartosc_tablicy_danerejestracji();
}
```

8.8. Rezultat działań – strona Rejestracje posiada metodę prerender do prezentacji danych z obiektów typu ApplicationBean1



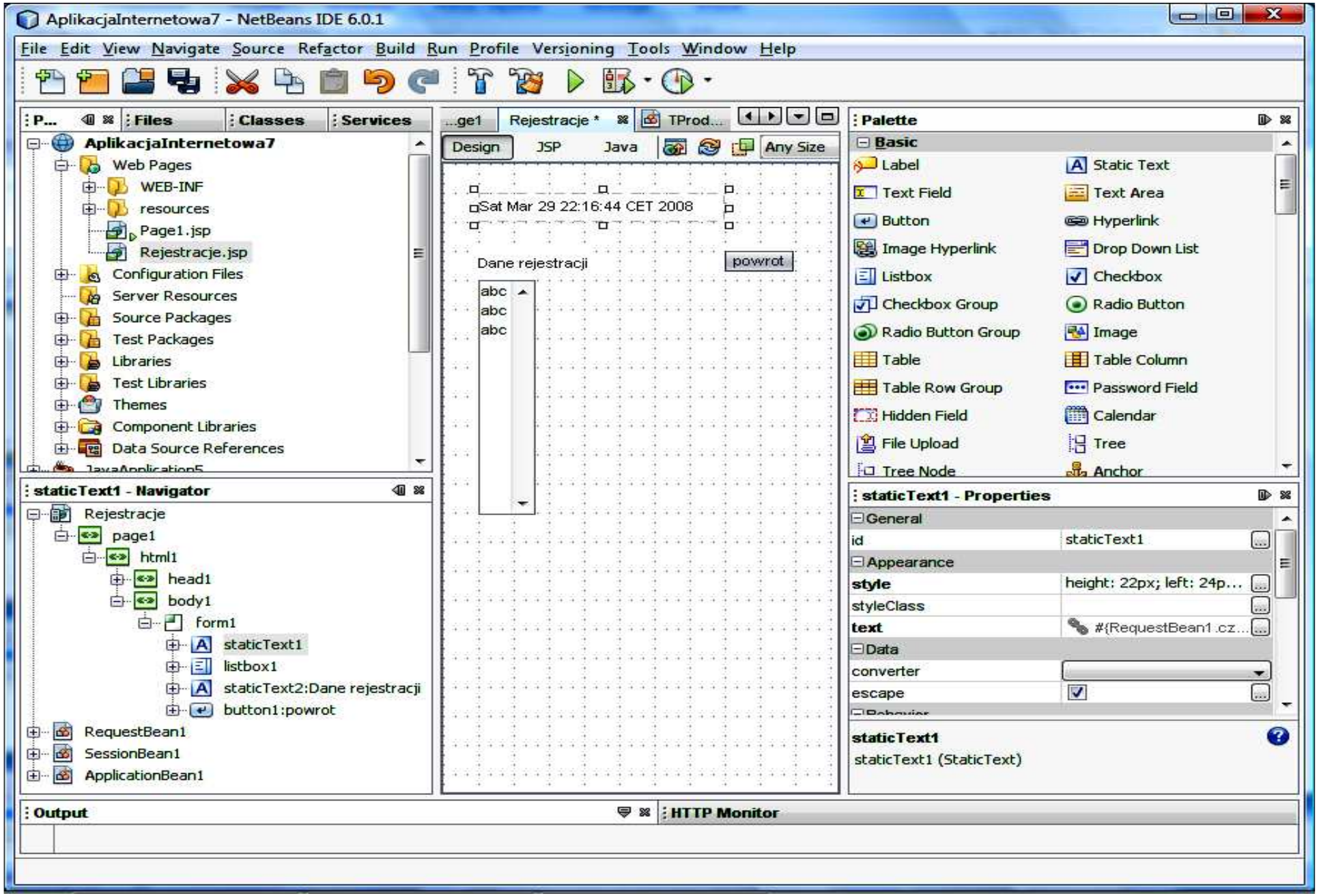
8.9. Tworzenie połączenia komponentu staticText1 typu Static Text ze strony Rejestracje.jsp z atrybutem czas z obiektu typu RequestBean1 – wywołanie opcji Bind to Data dla komponentu typu Static Text z wyskakującego menu



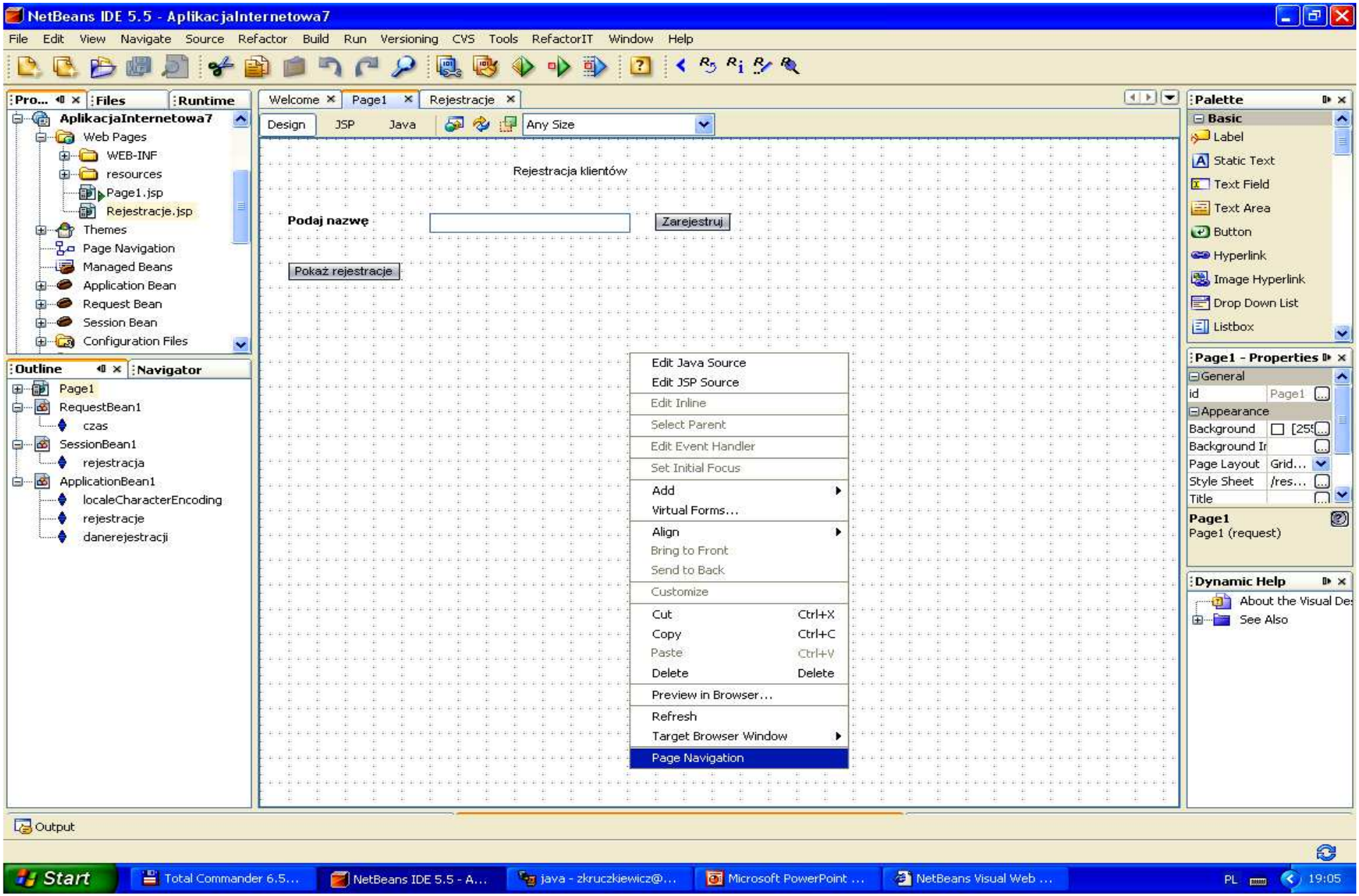


8.10. Połączenie komponentu typu Static Text z obiektem czas z obiektu typu RequestBean1 – czas będzie wyświetlał się tylko wtedy, gdy naciska się klawisz Zarejestruj na stronie Page1.jsp pozwalający przejść do strony Rejestracje.jsp. Obsługa zdarzenia tego przycisku (slajd 9.2 – metoda button1_action) tworzy nowy obiekt czas typu Date w obiekcie RequestBean1, przechowywany jedynie do ponownego wywołania tej strony, kiedy powstaje nowy obiekt typu RequestBean1. Po naciśnięciu przycisku Pokaż rejestracje po przejściu na stronę Rejestracja.jsp nie wyświetli się czas, gdyż w nowym obiekcie RequestBean1 nie utworzono obiektu czas podczas obsługi zdarzenia związanego z naciśnięciem tego przycisku (slajd 9.2 – metoda button2_action).

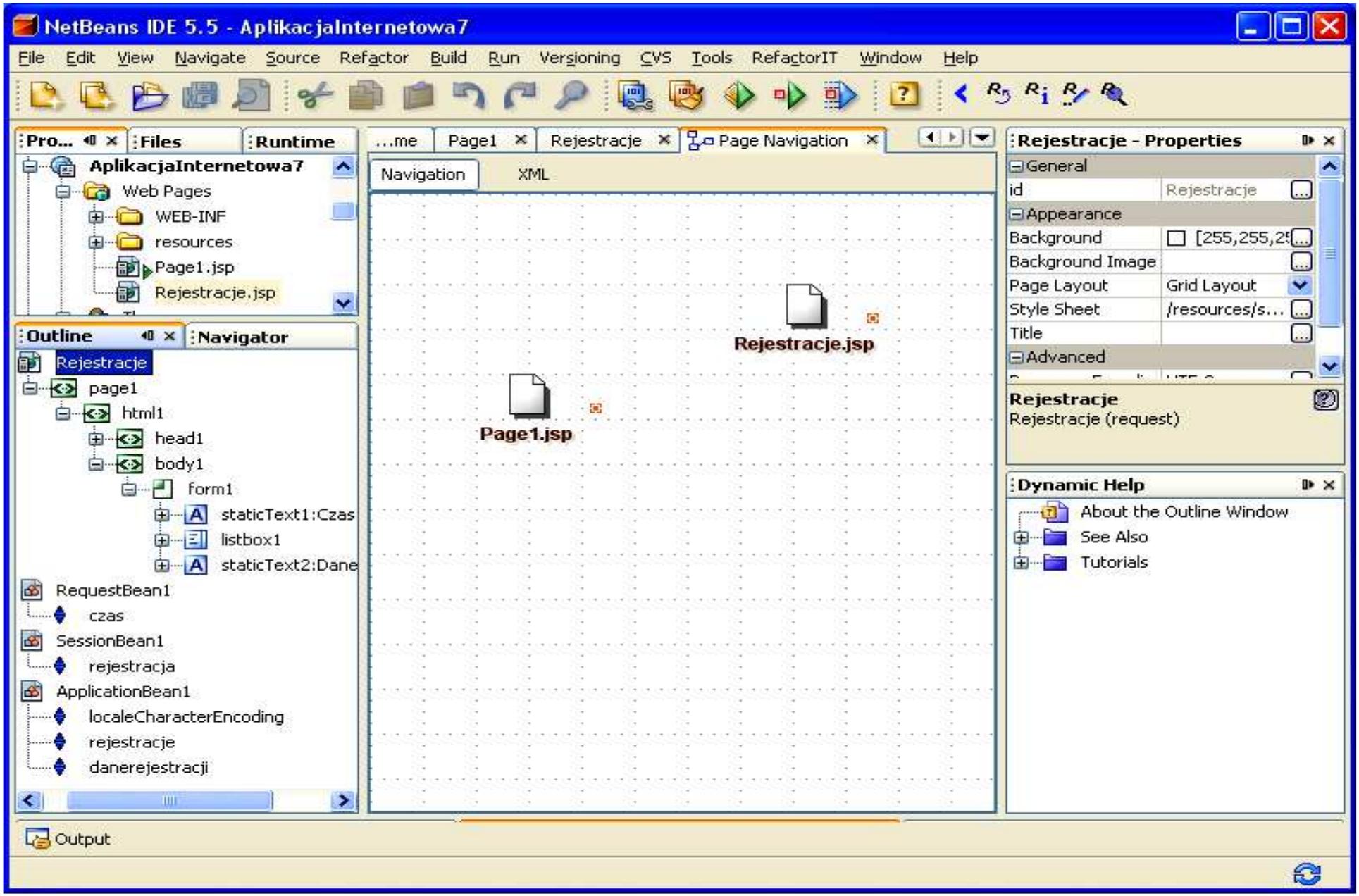
8.11. Efekt bindowania pola typu StaticText

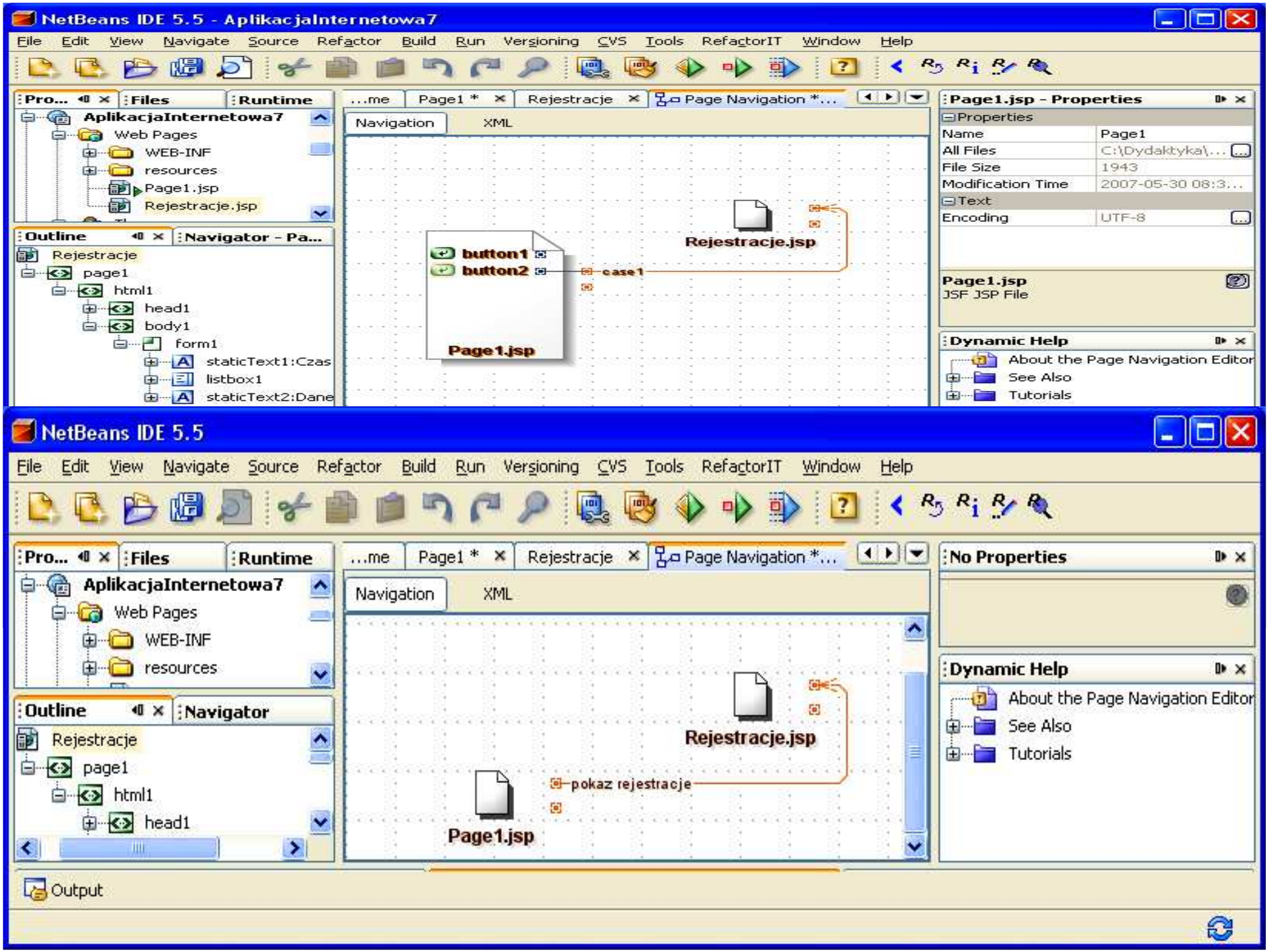


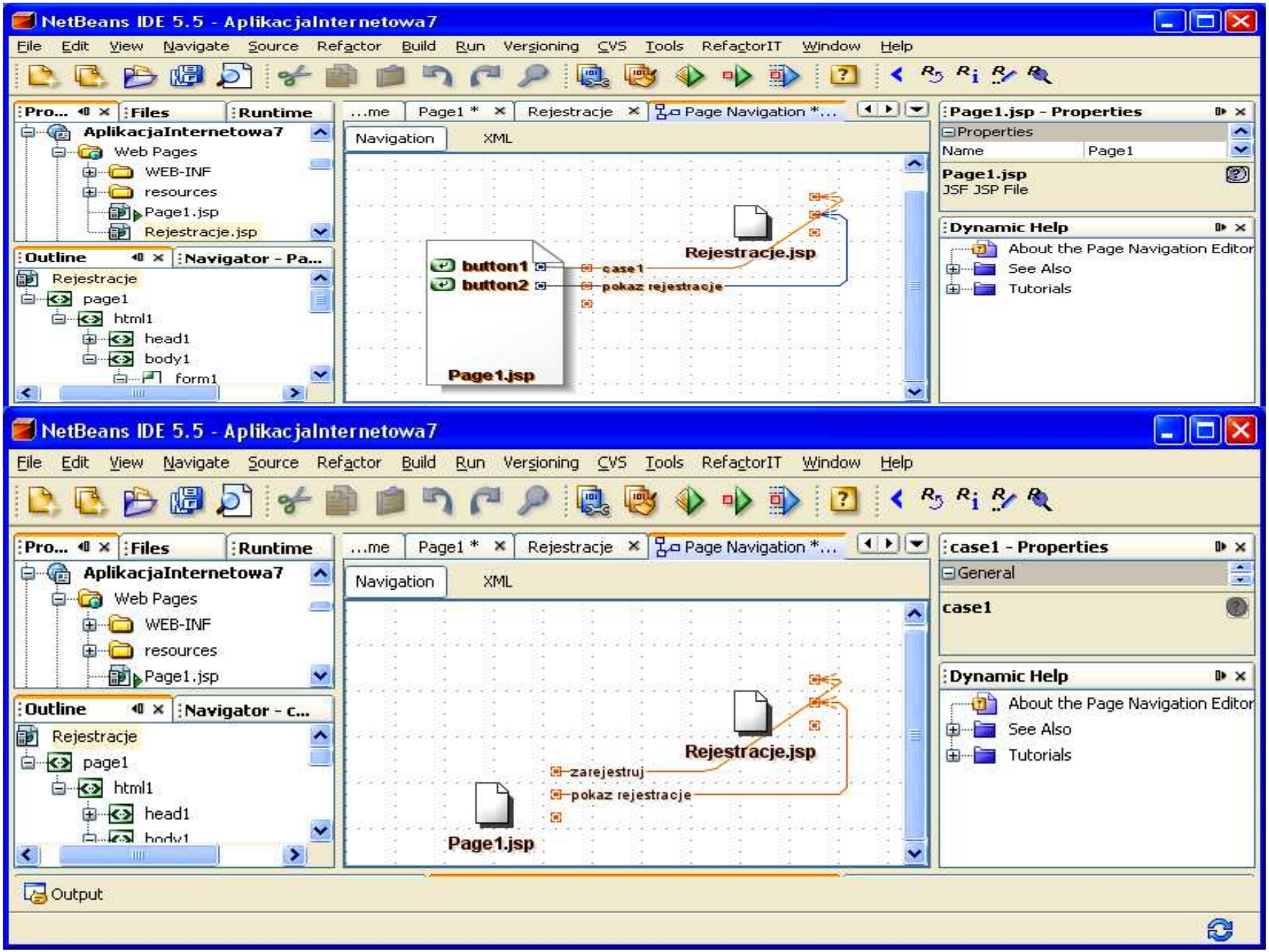
9. Wywołanie edytora Page Navigation z wyskakującego menu po naciśnięciu prawego klawisza myszy w obszarze Design strony



9.1. Utworzenie połączeń ze strony Page1 do strony Rezerwacje uruchamianych za pomocą klawiszy Pokaż rejestracje i Zarejestruj



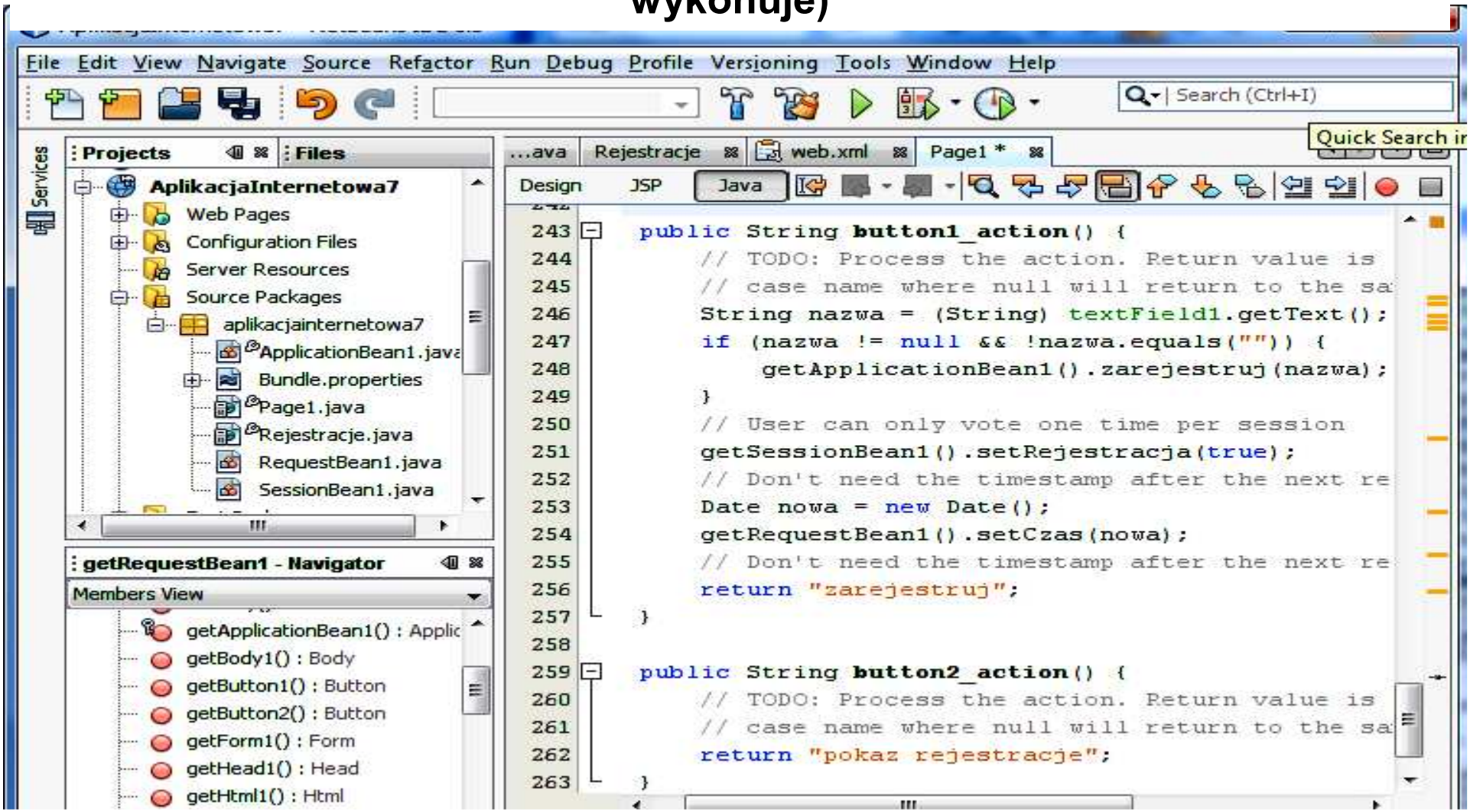

NetBeans IDE 5.5 - AplikacjaInternetowa7
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Pro... Files Runtime
AplikacjaInternetowa7
Web Pages
WEB-INF
resources
Page1.jsp
Rejestracje.jsp
Outline Navigator - Pa...
Rejestracje
page1
html1
head1
body1
form1
staticText1:Czas
listbox1
staticText2:Dane
...me Page1 * Rejestracje Page Navigation *...
Navigation XML
button1
button2
case1
Rejestracje.jsp
Page1.jsp
Page1.jsp - Properties
Properties
Name Page1
All Files C:\Dydaktyka\...
File Size 1943
Modification Time 2007-05-30 08:3...
Text
Encoding UTF-8
Page1.jsp
JSF JSP File
Dynamic Help
About the Page Navigation Editor
See Also
Tutorials
NetBeans IDE 5.5
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Pro... Files Runtime
AplikacjaInternetowa7
Web Pages
WEB-INF
resources
Outline Navigator
Rejestracje
page1
html1
head1
...me Page1 * Rejestracje Page Navigation *...
Navigation XML
Rejestracje.jsp
pokaz rejestracje
Page1.jsp
No Properties
Dynamic Help
About the Page Navigation Editor
See Also
Tutorials
Output


NetBeans IDE 5.5 - AplikacjaInternetowa7
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Pro... Files Runtime
AplikacjaInternetowa7
Web Pages
WEB-INF
resources
Page1.jsp
Rejestracje.jsp
Outline Navigator - Pa...
Rejestracje
page1
html1
head1
body1
form1
...me Page1 * Rejestracje Page Navigation *...
Navigation XML
button1
button2
case1
pokaz rejestracje
Rejestracje.jsp
Page1.jsp
Page1.jsp - Properties
Properties
Name Page1
Page1.jsp
JSF JSP File
Dynamic Help
About the Page Navigation Editor
See Also
Tutorials
NetBeans IDE 5.5 - AplikacjaInternetowa7
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Pro... Files Runtime
AplikacjaInternetowa7
Web Pages
WEB-INF
resources
Page1.jsp
Outline Navigator - c...
Rejestracje
page1
html1
head1
...me Page1 * Rejestracje Page Navigation *...
Navigation XML
zarejestruj
pokaz rejestracje
Rejestracje.jsp
Page1.jsp
case1 - Properties
General
case1
Dynamic Help
About the Page Navigation Editor
See Also
Tutorials
Output

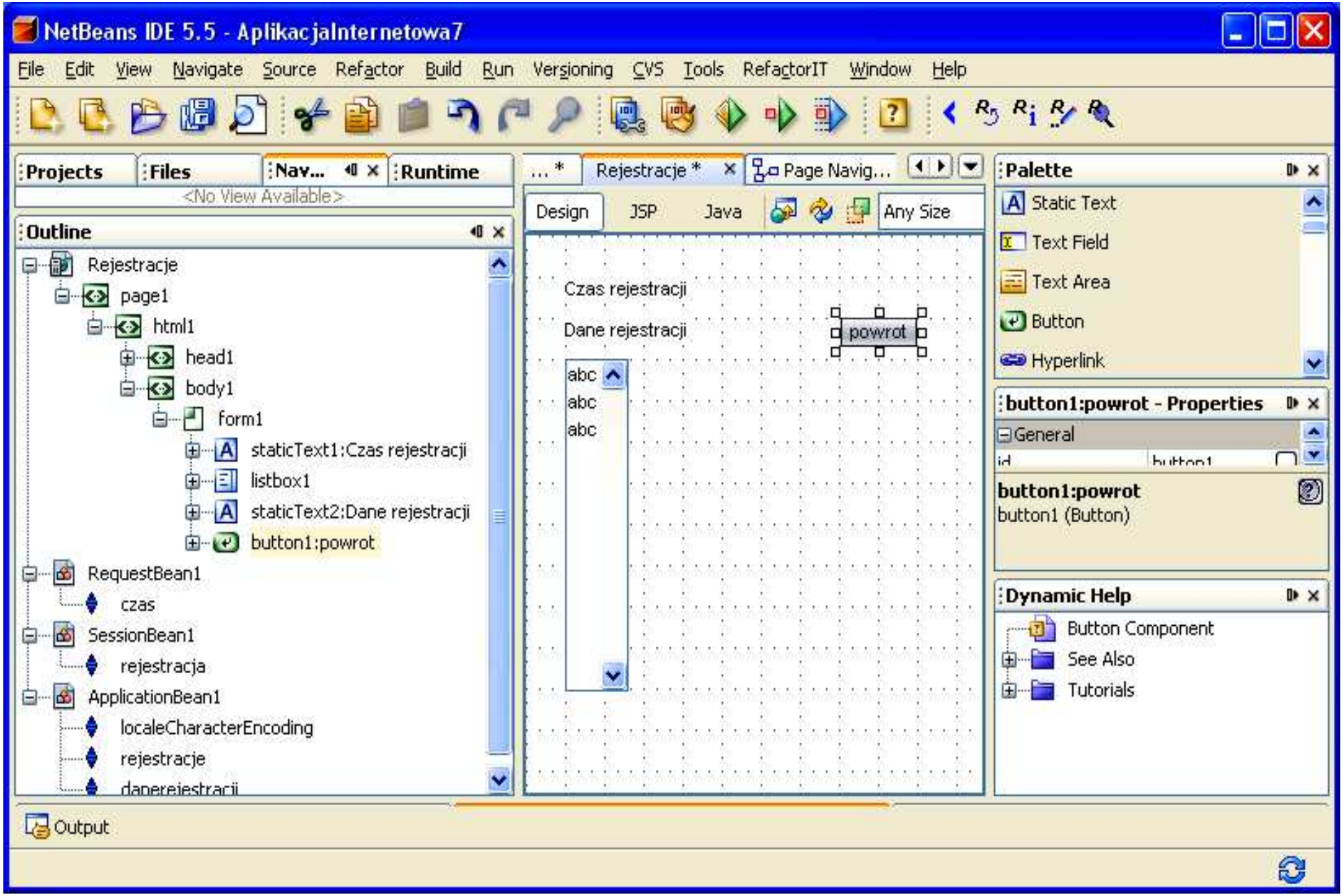
9.2. Obsługa zdarzeń wraz z obsługą połączeń strony Page1 ze stroną Rejestracje klawisza Zarejestruj (button1_action) oraz klawisza Pokaż rejestracje (button2_action)

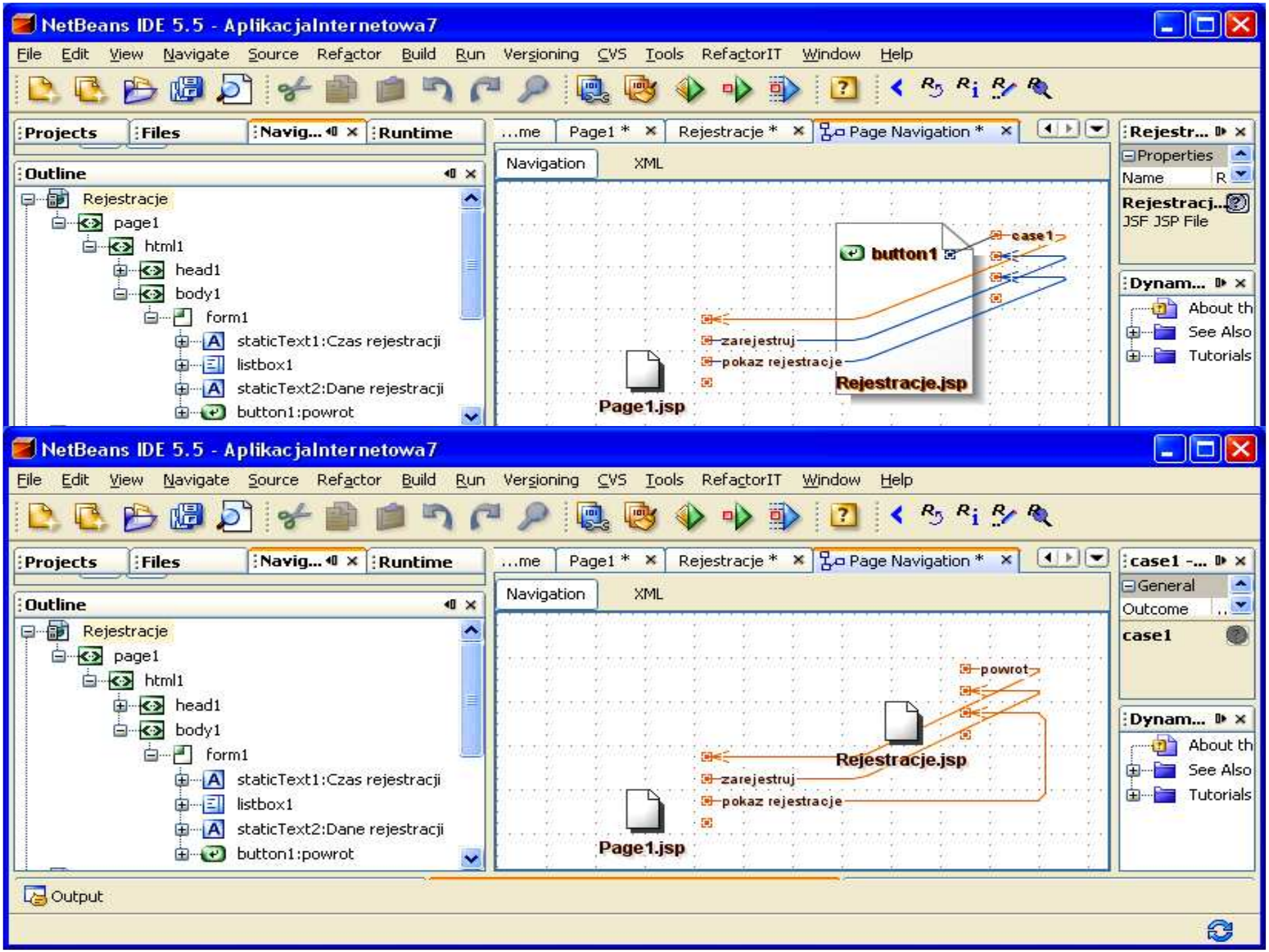
```
public String button1_action() {
        // TODO: Process the action. Return value is a navigation
        // case name where null will return to the same page.
    String nazwa = (String)textField1.getText();
     if (nazwa !=null && !nazwa.equals(""))
       getApplicationBean1().zarejestruj( nazwa);
            // User can only vote one time per session
     getSessionBean1().setRejestracja(true);
            // Don't need the timestamp after the next request ends
     Date nowa = new Date();
     getRequestBean1().setCzas(nowa);
            // Don't need the timestamp after the next request ends
       return "zarejestruj";      //obsługa polaczenia ze strona Rejestracje
    }

public String button2_action() {
       // TODO: Process the action. Return value is a navigation
    // case name where null will return to the same page.
     return "pokaz rejestracje"; //obsługa polaczenia ze strona Rejestracje
   }
```

9.3. Rezultat działań – strona Page1 posiada metody obsługi zdarzeń przycisków Zarejestruj (metoda button1_action podczas połączenia ze stroną Rejestracje realizuje rejestrację klienta oraz wyłącza przycisk Zarejestruj oraz ustawia czas rejestracji) i Pokaż rejestracje (metoda button2_action podczas połączenia ze stroną Rejestracje nic nie wykonuje)

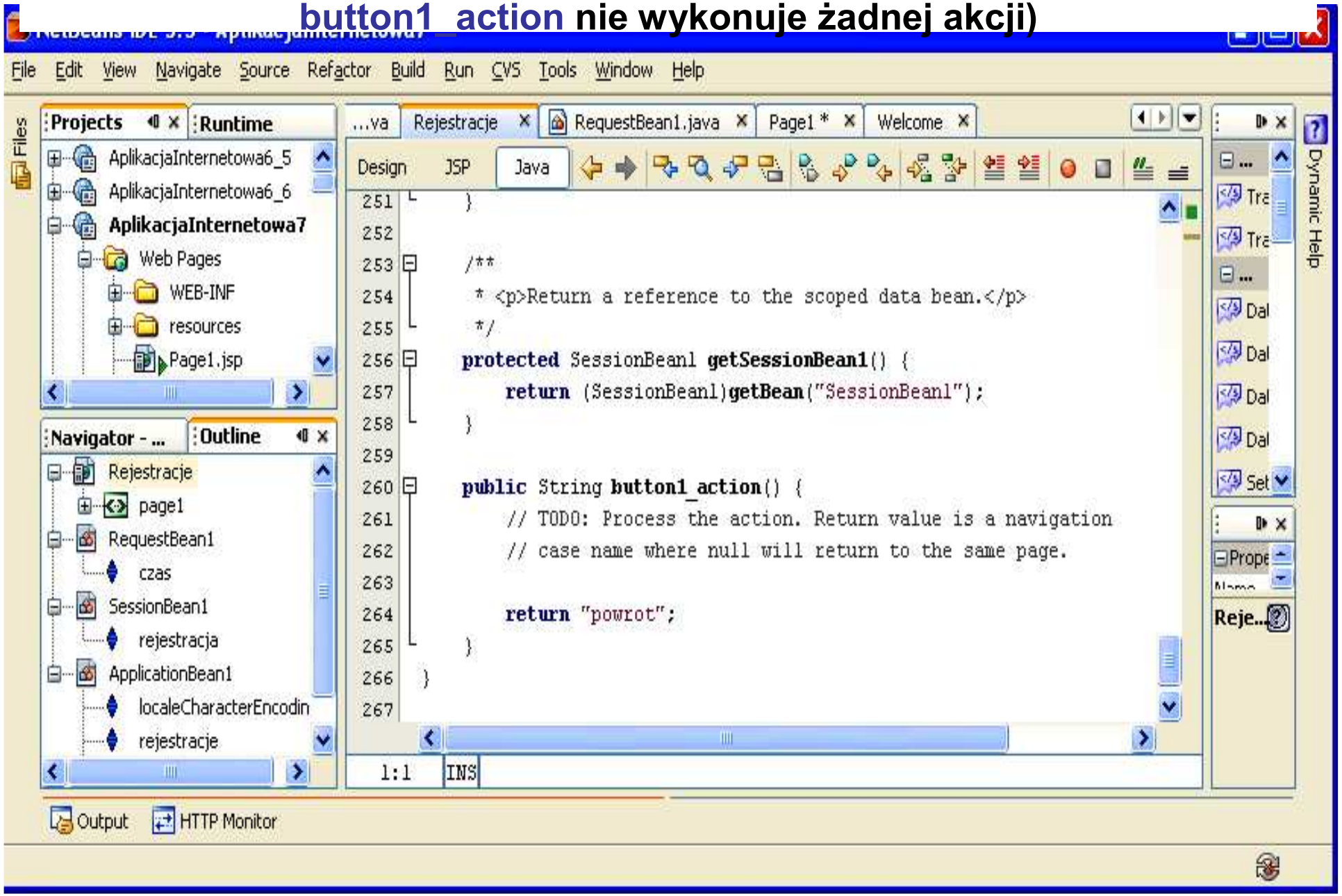


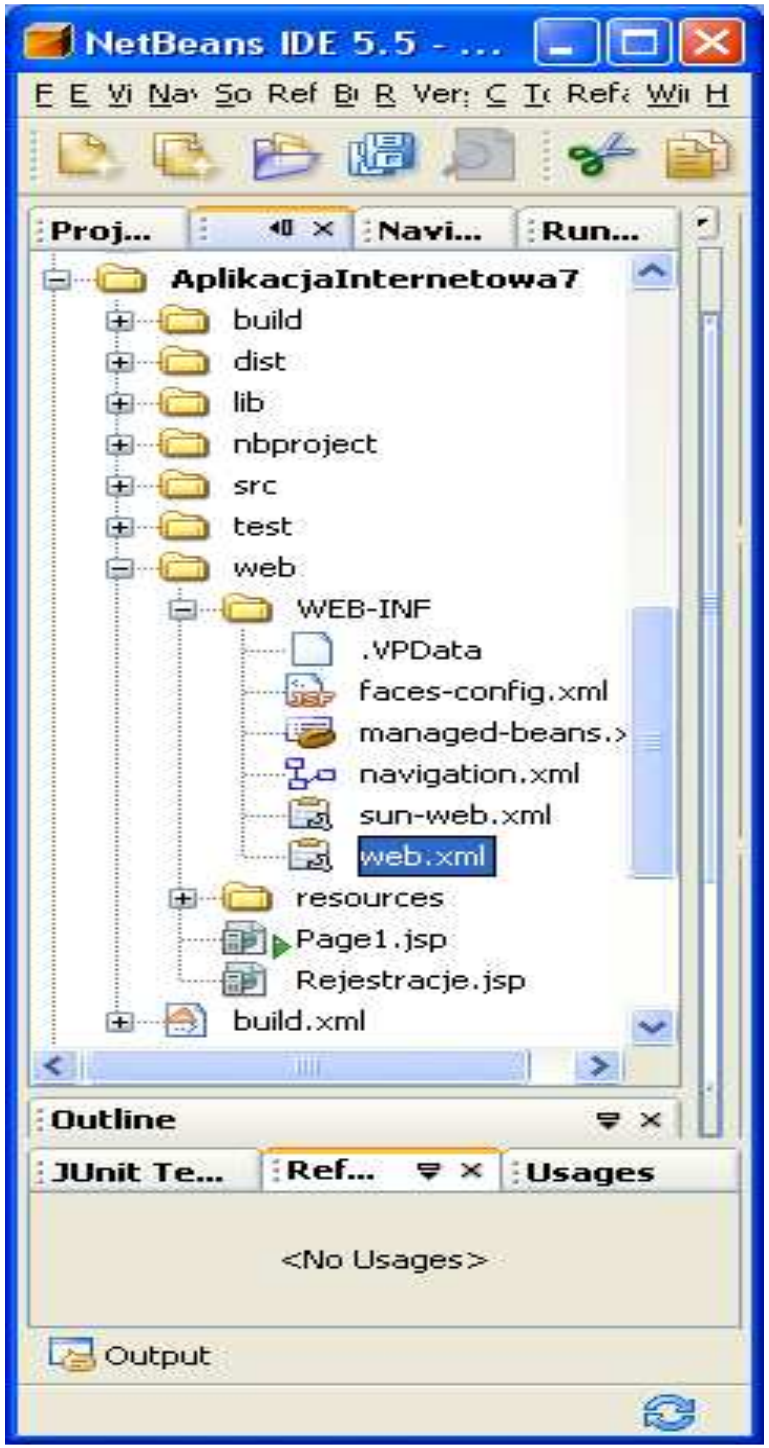
9.4. Obsługa klawisza powrot – wstawienie połączenia do strony Page1 ze strony Rejestracje




NetBeans IDE 5.5 - AplikacjaInternetowa7
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Projects Files Navig... Runtime
Outline
Rejestracje
page1
html1
head1
body1
form1
staticText1:Czas rejestracji
listbox1
staticText2:Dane rejestracji
button1:powrot
...me Page1 * Rejestracje * Page Navigation *
Navigation XML
button1
case1
zarejestruj
pokaz rejestracje
Rejestracje.jsp
Page1.jsp
Rejestr...
Properties
Name R
Rejestracj...
JSF JSP File
Dynam...
About th
See Also
Tutorials
NetBeans IDE 5.5 - AplikacjaInternetowa7
File Edit View Navigate Source Refactor Build Run Versioning CVS Tools RefactorIT Window Help
Projects Files Navig... Runtime
Outline
Rejestracje
page1
html1
head1
body1
form1
staticText1:Czas rejestracji
listbox1
staticText2:Dane rejestracji
button1:powrot
...me Page1 * Rejestracje * Page Navigation *
Navigation XML
powrot
Rejestracje.jsp
zarejestruj
pokaz rejestracje
Page1.jsp
case1 -...
General
Outcome
case1
Dynam...
About th
See Also
Tutorials
Output

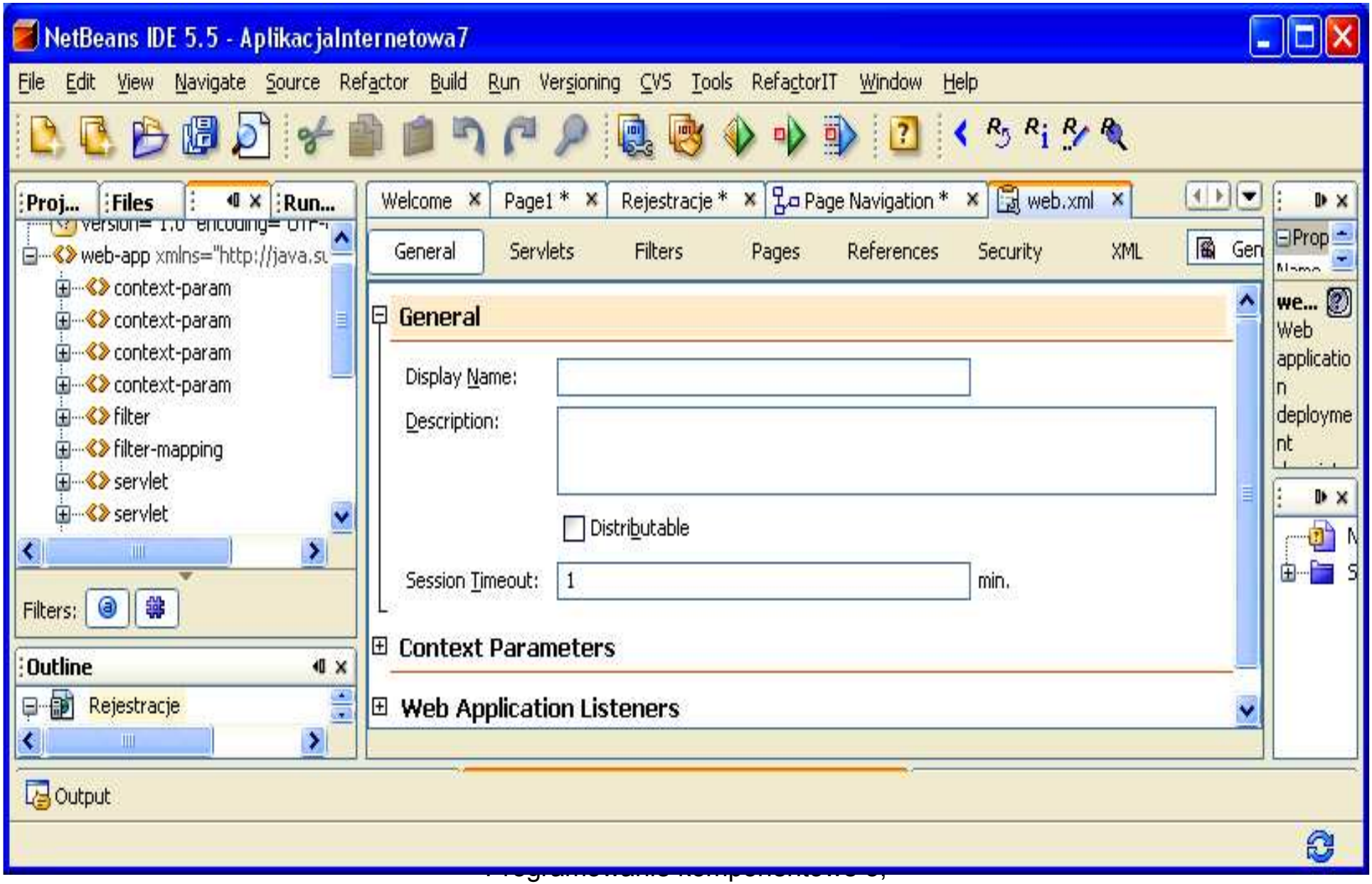
9.5. Obsługa połączenia ("powrot") strony Rejestracje ze stroną Page1 za pomocą klawisza powrot (oprócz połączenia ze stroną Page1 metoda button1_action nie wykonuje żadnej akcji)



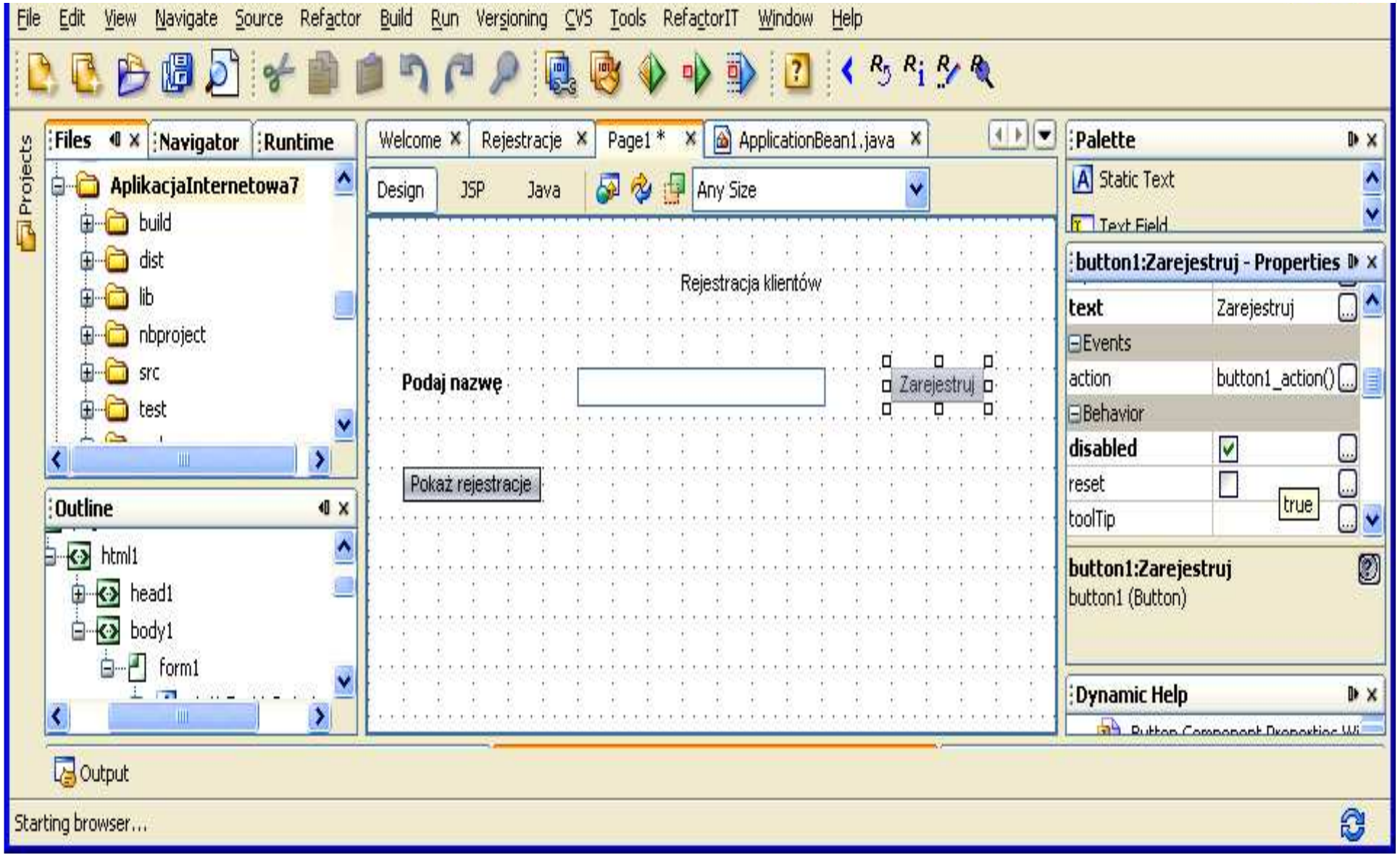


10. Ustawienie czasu typu Timeout dla obiektu typu Sessionbean1 w pliku web.xml

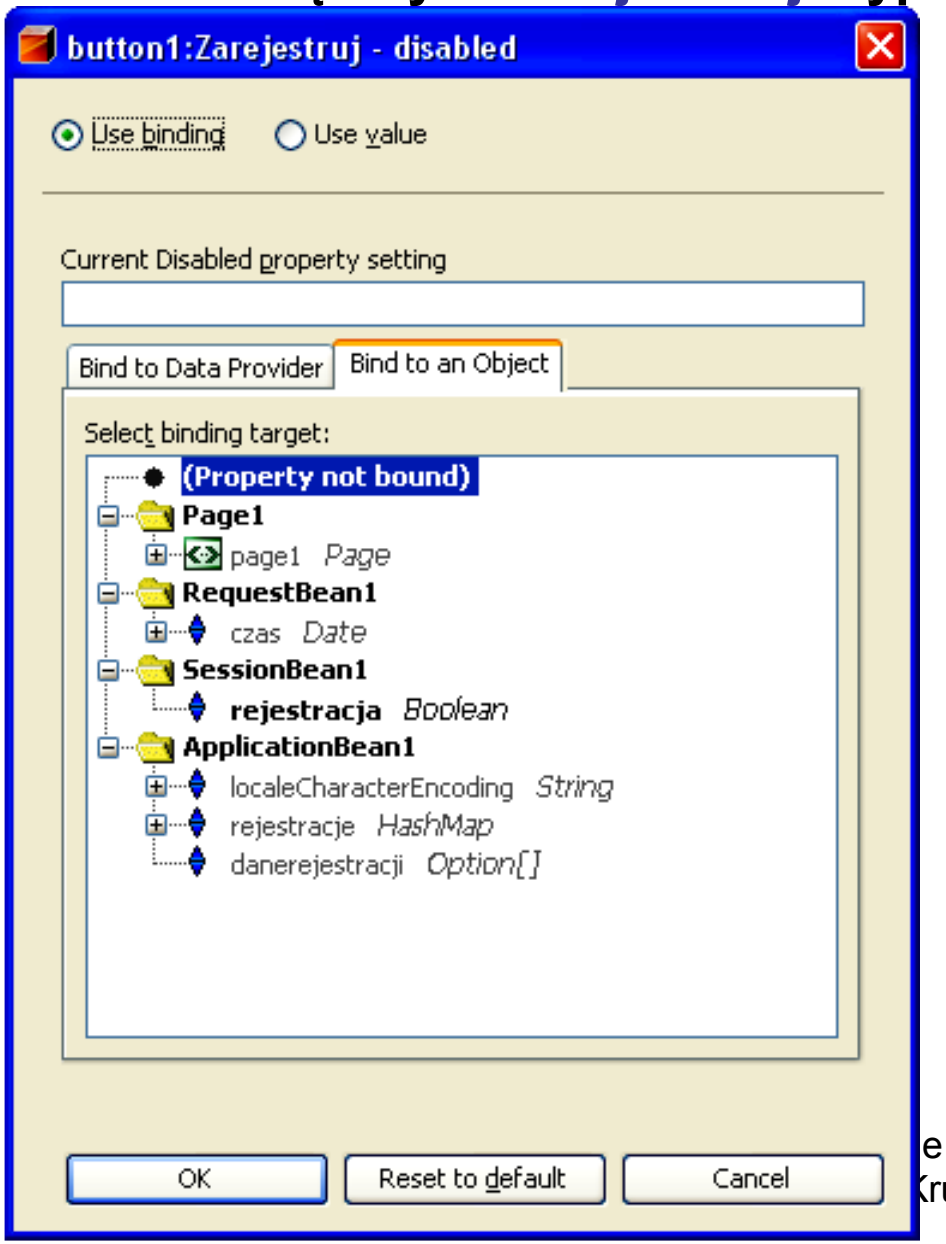
10.1. Formularz pliku web.xml

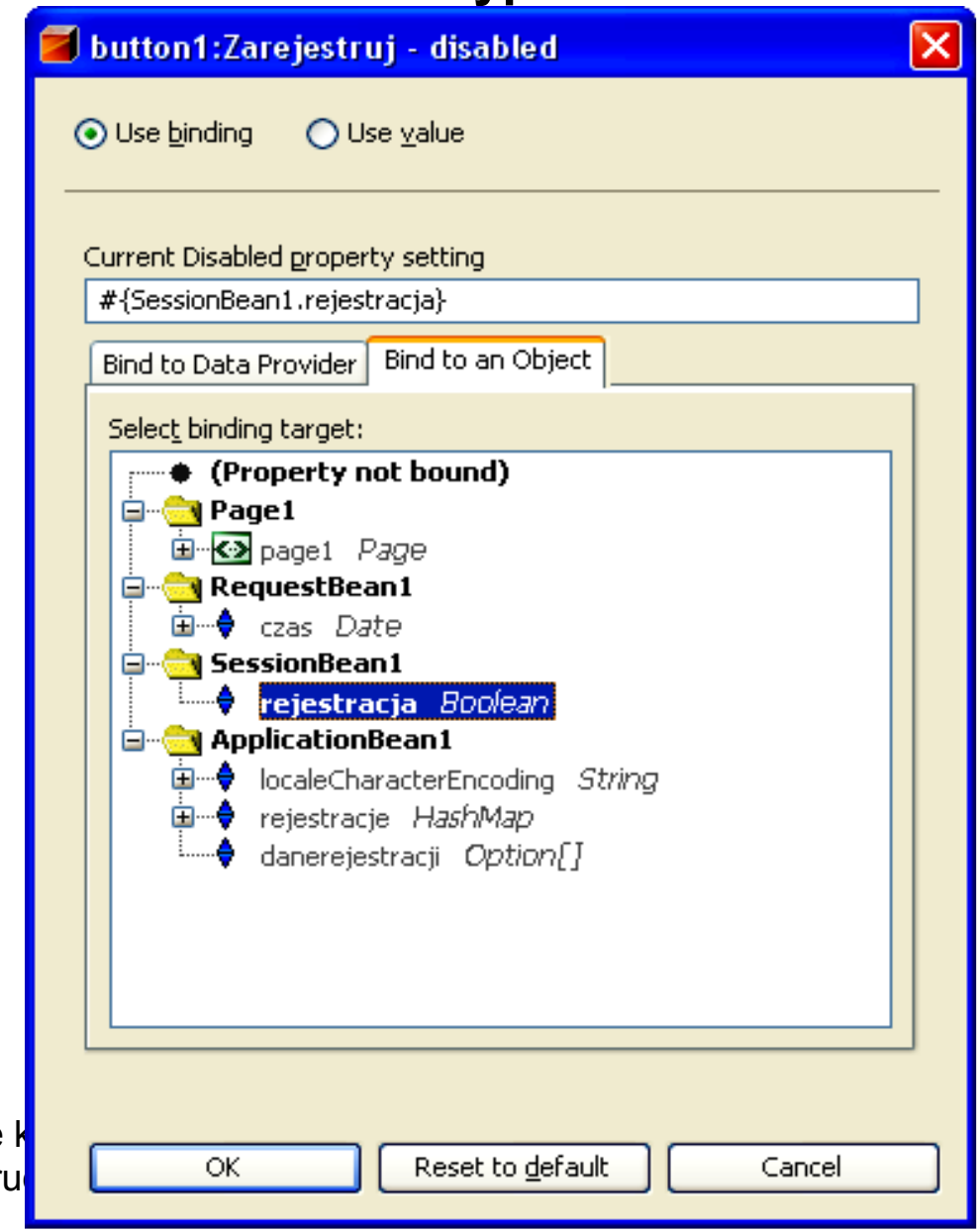


11. Ustawianie właściwości disabled klawisza Zarejestruj w celu zablokowania wywołania zdarzenia rejestracji za pomocą metody button1_action– ustawienie klawisza checkbox oraz wywołanie edytora tej właściwości

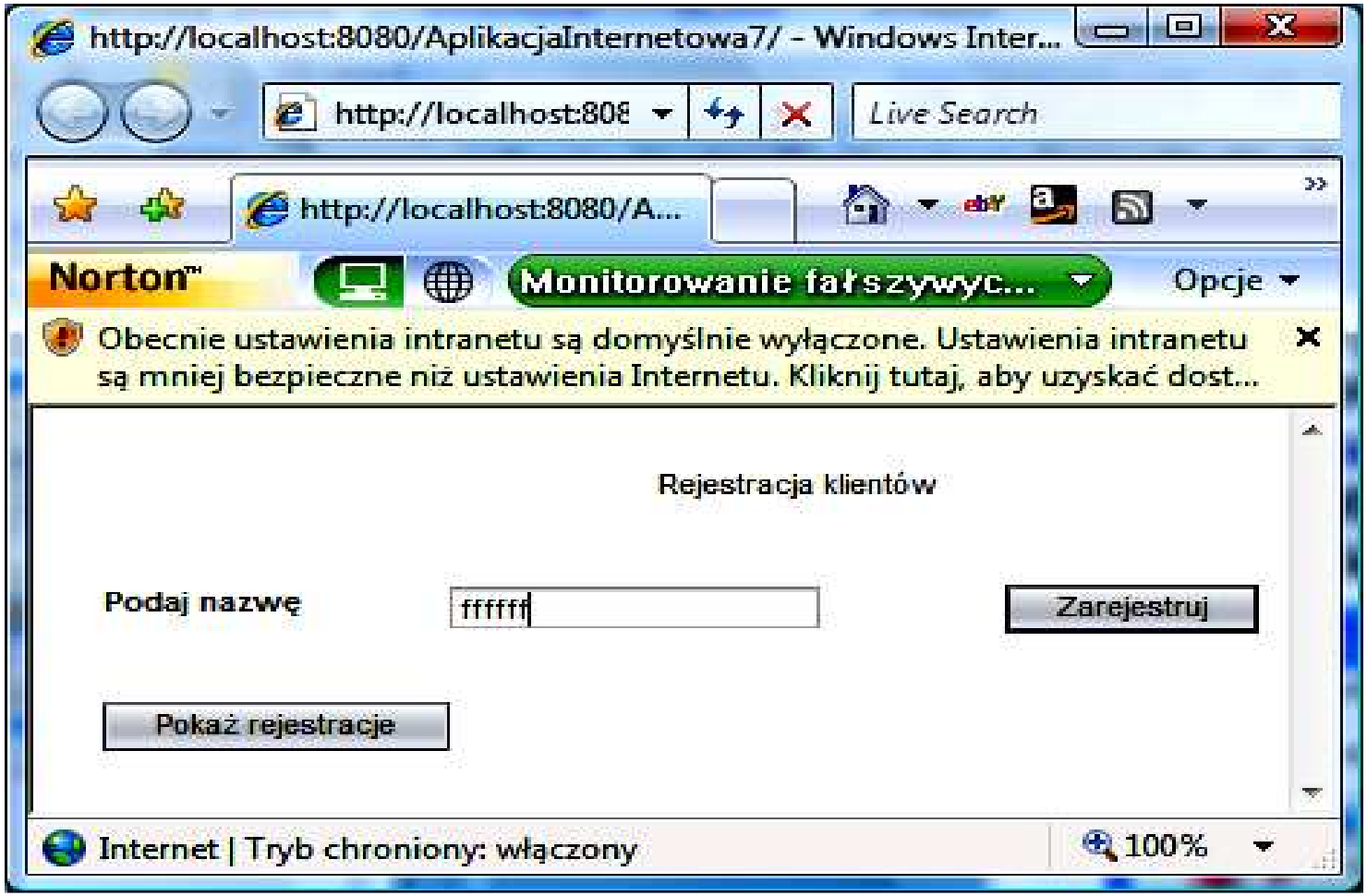


11.1. Edytor właściwości disabled klawisza Zarejestruj - ustawienie właściwości klawisza typu disabled w zakładce Bind to an Object zgodnie z wartością atrybutu rejestracja typu boolean obiektu typu SessionBean1

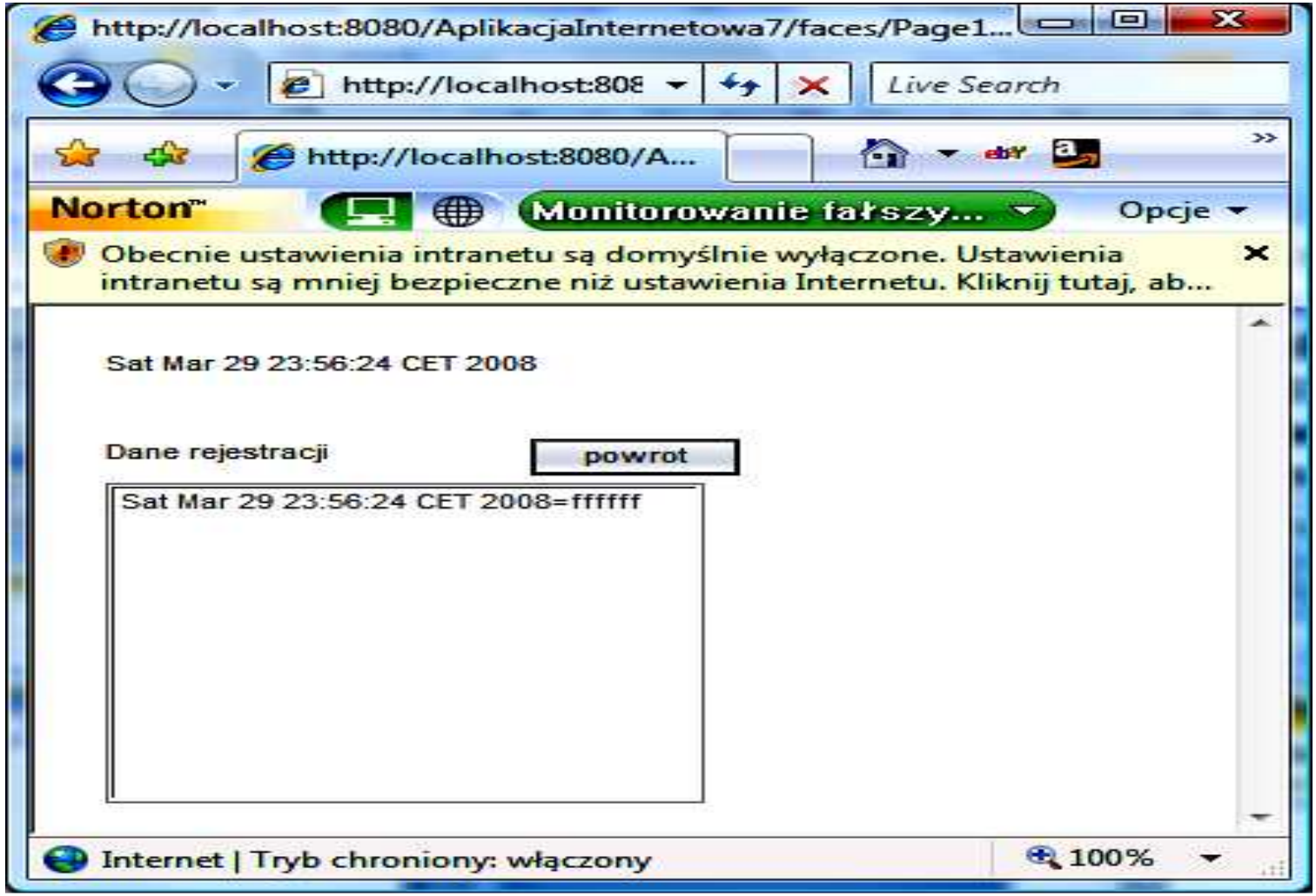




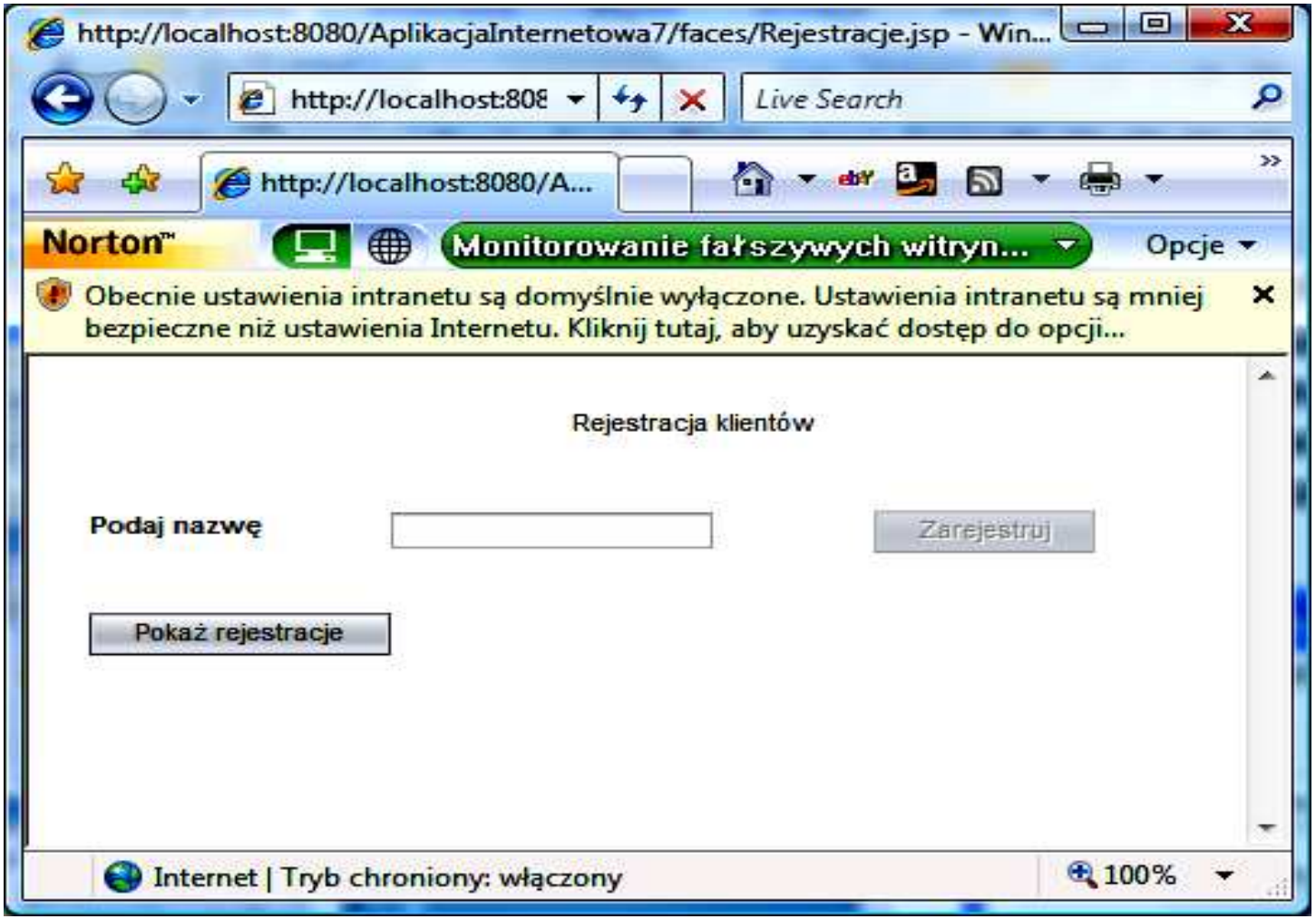
12. Stan strony głównej po wywołaniu strony głównej aplikacji – klawisz Pokaż rejestracje realizuje połączenie ze stroną Rejestracje, która prezentuje rejestracje klientów wykonanych podczas działania całej aplikacji. Klawisz Zarejestruj wywołuje obsługę rejestracji klienta podczas realizacji połączenia ze stroną Rejestracje, która prezentuje rejestracje klientów wykonanych podczas działania całej aplikacji i jednocześnie czas ostatniej rejestracji.



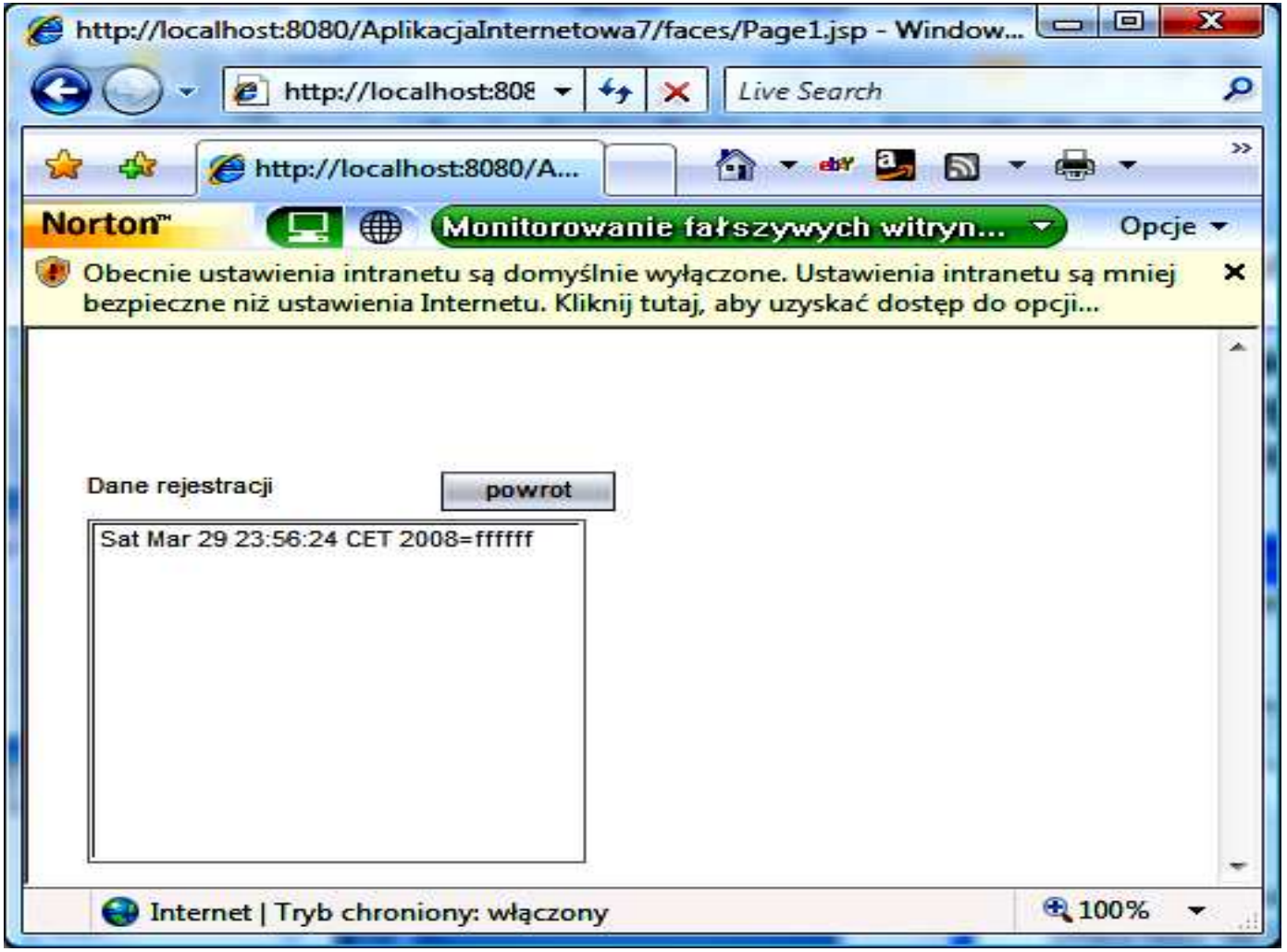
12.1. Stan po naciśnięciu klawisza Zarejestruj na stronie Page1, który wywołuje stronę Rejestracje, przekazuje zawartość atrybutu danerejestracji z danymi rejestracji z obiektu typu ApplicationBean1 za pomocą komponentu typu ListBox oraz wartość atrybutu czas z obiektu typu RequestBean1 (czas ostatniej rejestracji), które są wyświetlane w komponentach strony typu List Box oraz StaticText



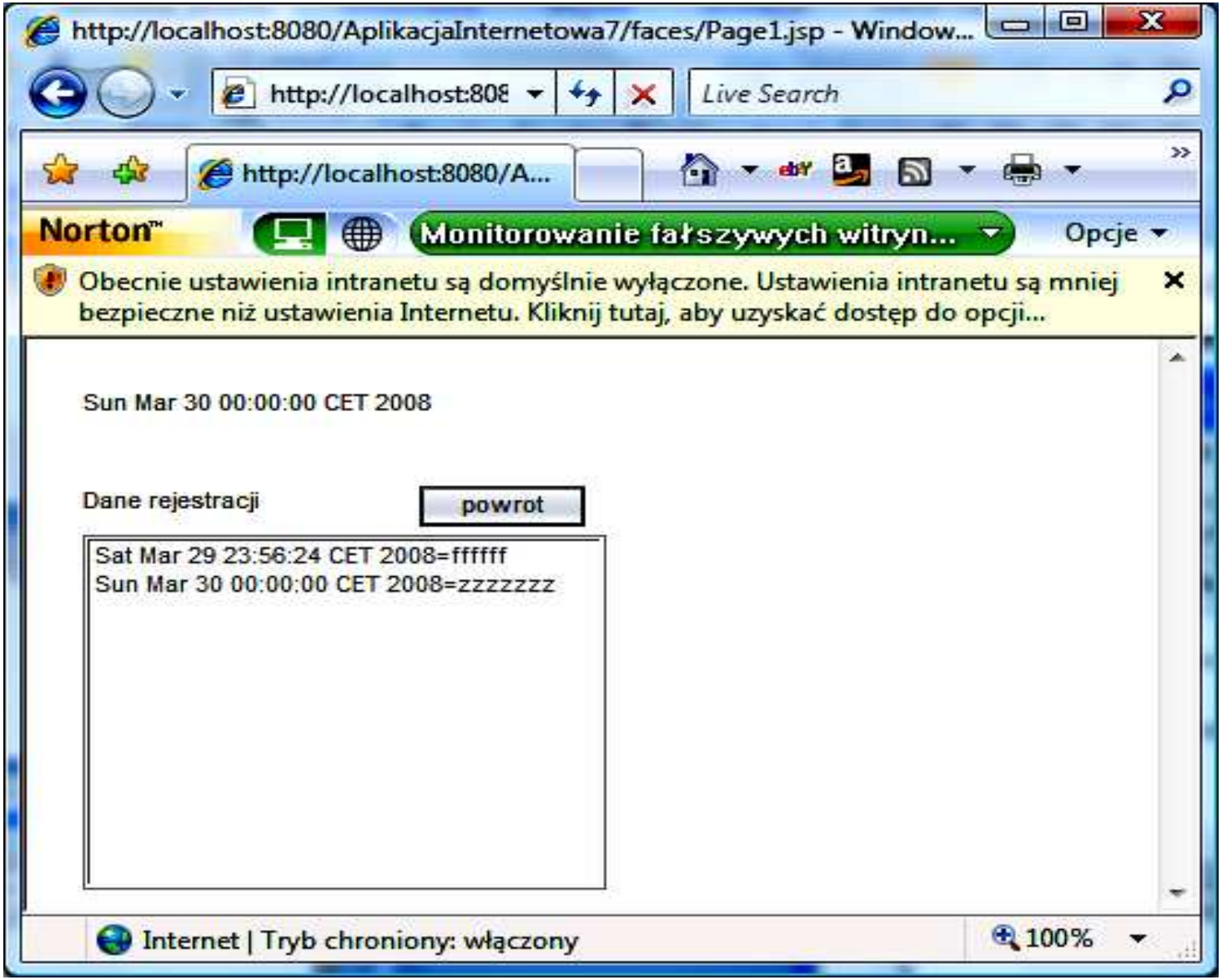
12.2. Stan zablokowania rejestracji z powodu stanu Disable klawisza Zarejestruj, który trwa od momentu wykonania rejestracji i połączenia się ze stroną Rejestracje do czasu zakończenia bieżącej instancji obiektu SessionBean1 po upływie czasu wyznaczonego przez Timeout (1min) - kończy się w chwili utworzenia nowego obiektu typu SessionBean1



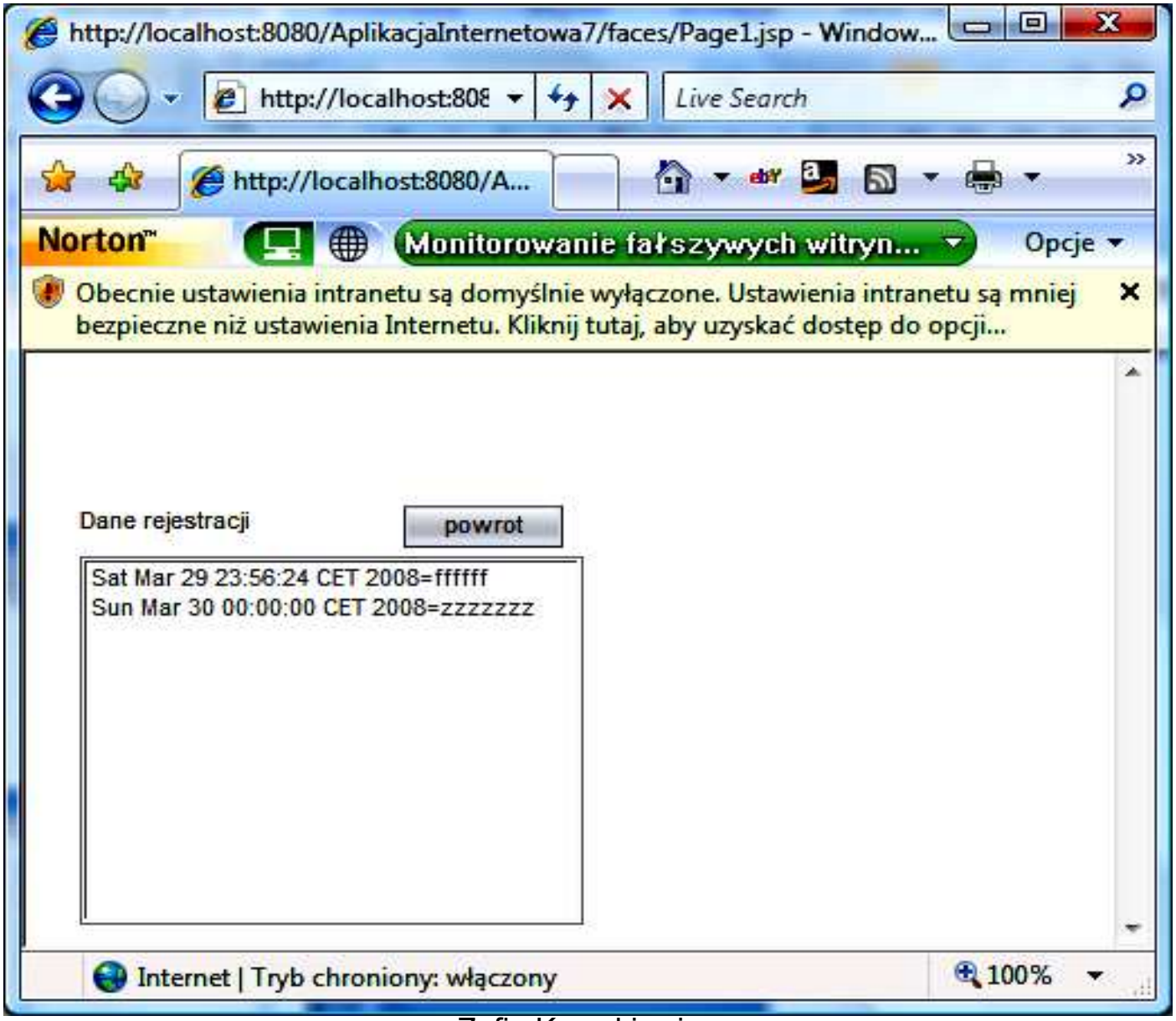
12.3. Wywołanie strony Rejestracje za pomocą klawisza Pokaż rejestracje. Dane atrybutu czas nie można już wyświetlić, ponieważ istnieje już inny egzemplarz obiektu RequestBean1, w którym nie ma już zapisanego czasu ostatniej rejestracji w atrybucie czas w momencie obsługi klawisza Zarejestruj (czyli realizacji fazy request po połączeniu ze stroną Rejestracje)

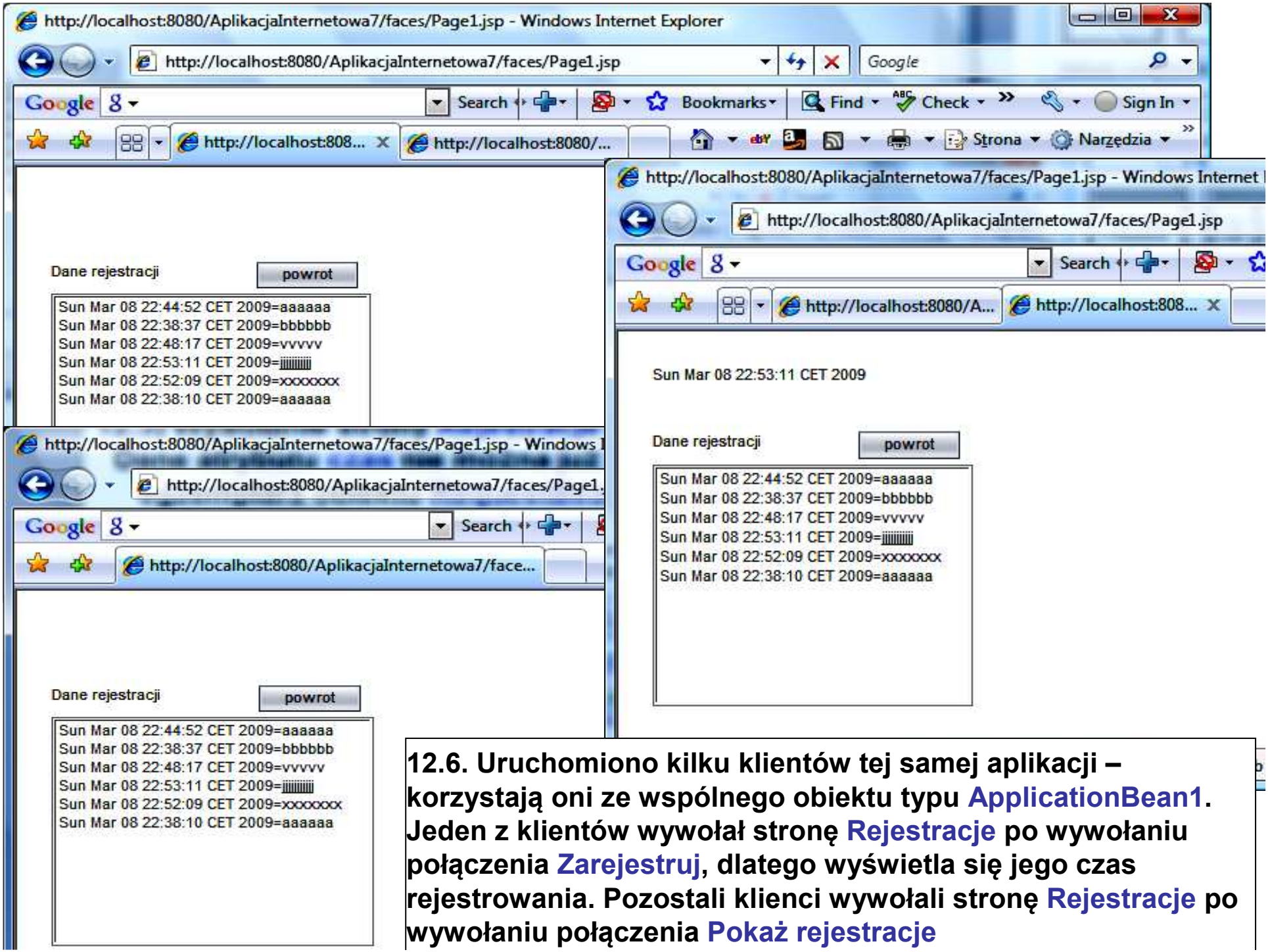


12.4. Ponowna obsługa klawisza Zarejestruj w momencie utworzenia nowych obiektów typu SessionBean1 oraz typu RequestBean1

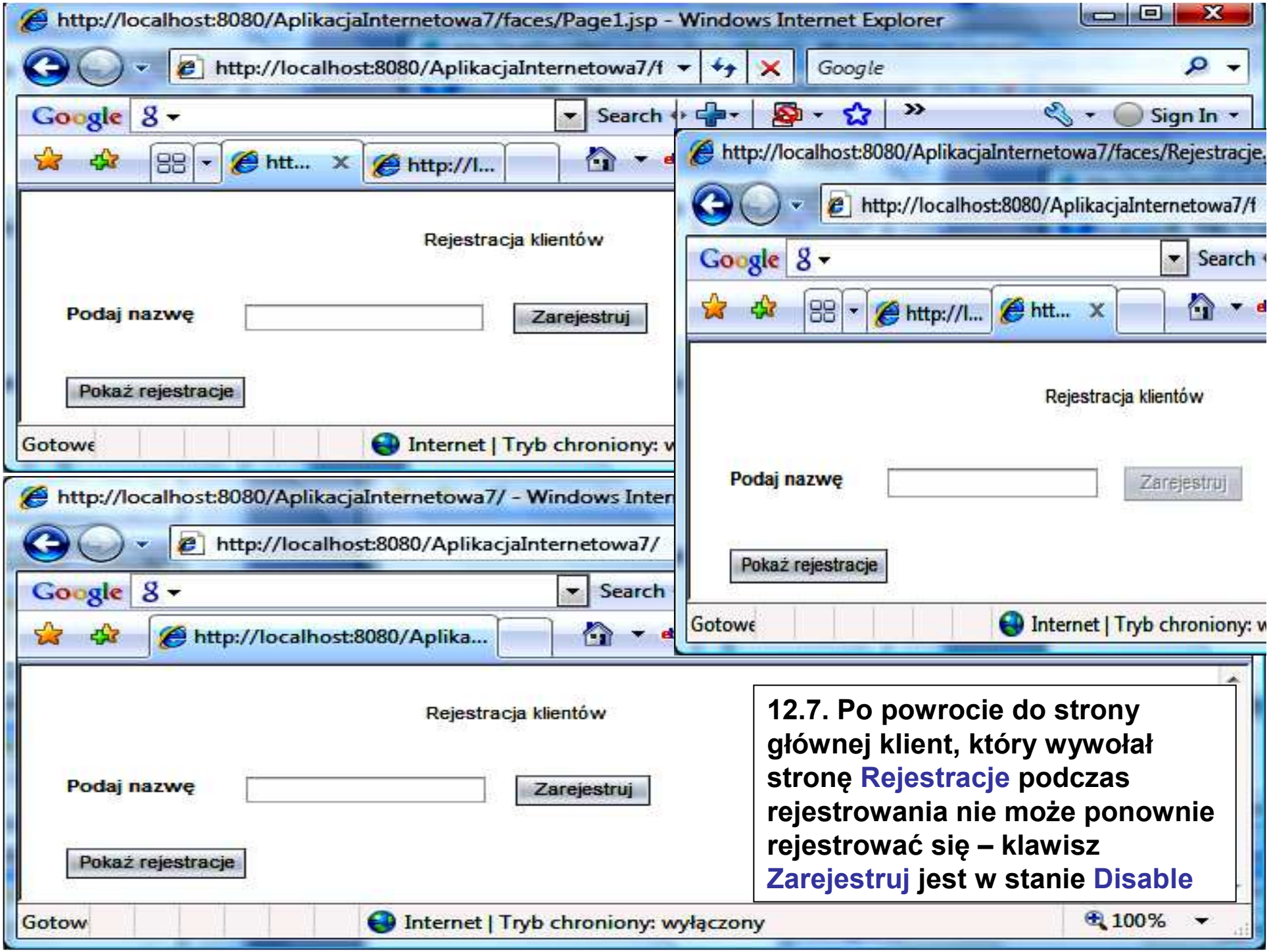


12.5. Ponowna obsługa klawisza Pokaz rejestracje w momencie utworzenia nowego obiektu typu RequestBean1





12.6. Uruchomiono kilku klientów tej samej aplikacji – korzystają oni ze wspólnego obiektu typu ApplicationBean1. Jeden z klientów wywołał stronę Rejestracje po wywołaniu połączenia Zarejestruj, dlatego wyświetla się jego czas rejestrowania. Pozostali klienci wywołali stronę Rejestracje po wywołaniu połączenia Pokaż rejestracje



12.7. Po powrocie do strony głównej klient, który wywołał stronę Rejestracje podczas rejestrowania nie może ponownie rejestrować się – klawisz Zarejestruj jest w stanie Disable