

**Utrwalanie danych – zastosowanie
obiektowego modelu danych warstwy
biznesowej do generowania schematu
relacyjnej bazy danych
(podrozdziały 1, 2, 3)**

**Tworzenie warstwy prezentacji – pierwszy
etap (podrozdziały 4, 5)**

Przykład

Autor

Zofia Kruczkiewicz

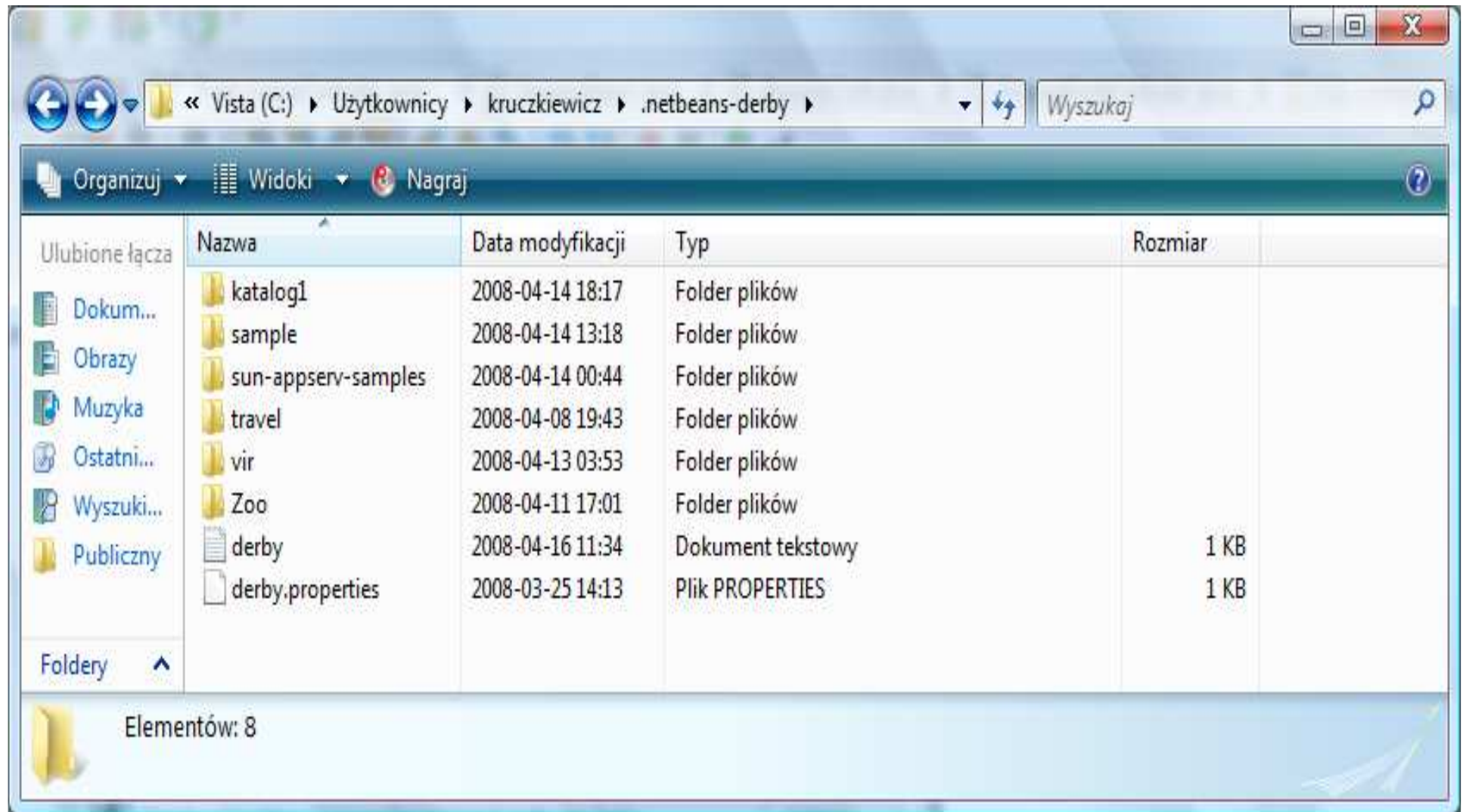
Wzorce oprogramowania -
laboratorium5_6

1. Tworzenie bazy danych w systemie baz danych Derby

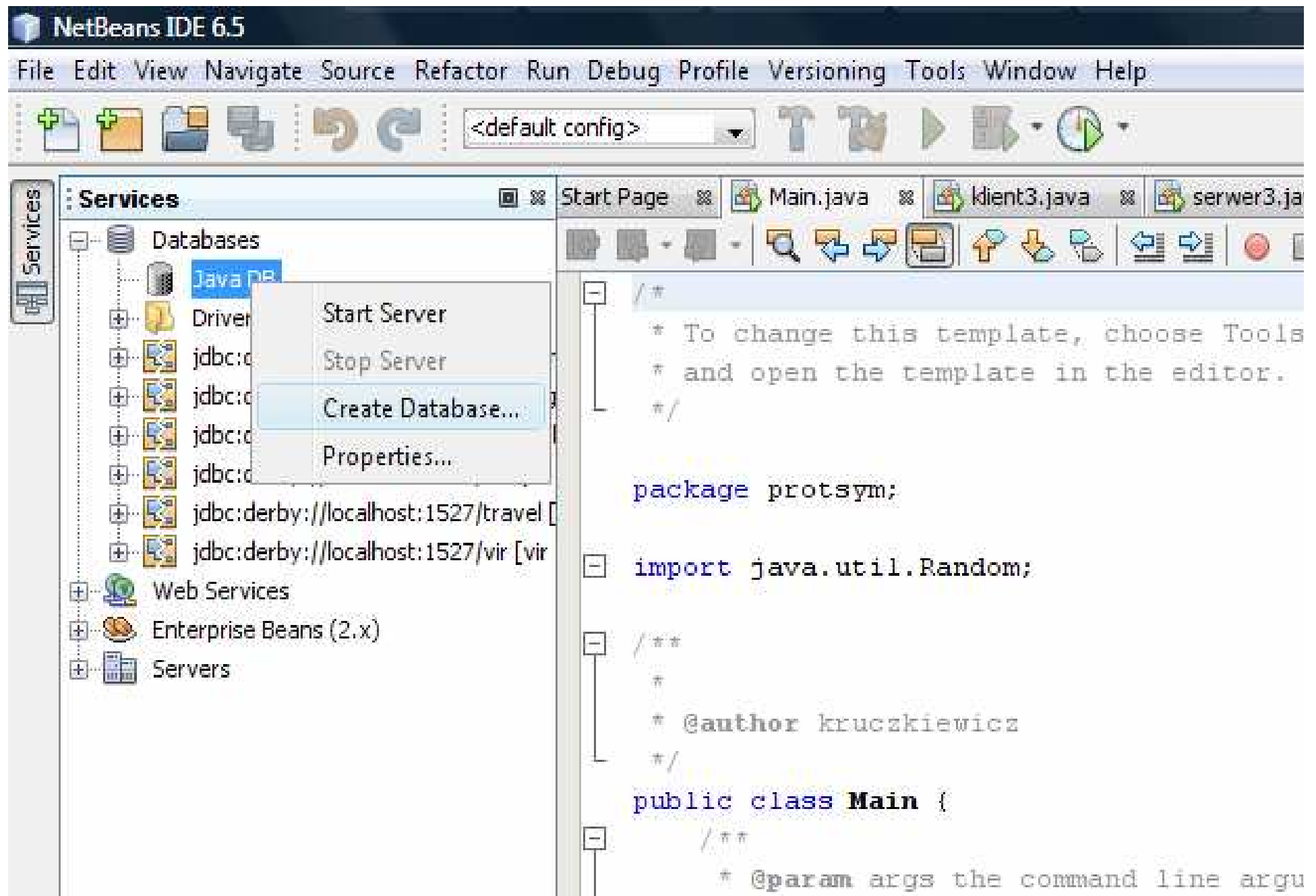
Tworzenie bazy danych w systemie baz danych Derby

- Należy wybrać katalog na pustą bazę danych w systemie bazy danych Derby – domyślnym katalogiem jest katalog użytkowników systemu Windows
\.netbeans-derby - slajd(1)
- Należy wybrać z zakładki **Services** środowiska NetBeans 6.5 opcję **Databases**. Następnie kliknąć prawym klawiszem myszy na **Java DB** i wybrać opcję **Create Database...** slajd (2)
- W formularzu tworzenia bazy danych wg wzoru podanego na slajdzie (3) wprowadzić dane dotyczące bazy danych: **Database Name, User, Password** oraz **Database Location** (domyślne lub wybrane przez użytkownika) – w wybranym katalogu pojawi się katalog o nazwie bazy danych z pustą bazą danych (slajd (4))
- W zakładce **Service** należy otworzyć opcję **Databases** (slajd (5)) i wybierając prawym klawiszem myszy pozycję z listy połączeń do baz danych typu **jdbc:derby://localhost:1527/nazwa_bazy_danych:[użytkownik on schematbazydanych]**, należy nacisnąć **Connect**. Po chwili nastąpi połączenie z bazą danych – pustą – symboliczny kwadrat z lewej strony łańcucha połączenia ulegnie scaleniu. Przy połączeniu domyślnie zapamiętane jest hasło do bazy danych. Można to zmienić w okienku **Properties** (wybór po kliknięciu prawym klawiszem myszy na pozycję połączenia)
- Należy kliknąć na symbol **[+]** z lewej strony łańcucha połączenia – pojawi się zawartość katalogu z trzema pustymi podkatalogami: **Tables, Views, Procedures** (slajd (6))

1.1. Zakładanie nowej bazy danych w domyślnym katalogu dla baz danych typu Derby (1)



Zakładanie pustej bazy danych dla systemu baz danych Derby (2)



NetBeans IDE 6.5

File Edit View Navigate Source Refactor Run Debug Profile Versioning Tools Window Help

<default config>

Services

Databases

- Java DB
 - Start Server
 - Stop Server
 - Create Database...
 - Properties...
- Driver
- jdbc:...
- jdbc:...
- jdbc:...
- jdbc:...
- jdbc:derby://localhost:1527/travel
- jdbc:derby://localhost:1527/vir

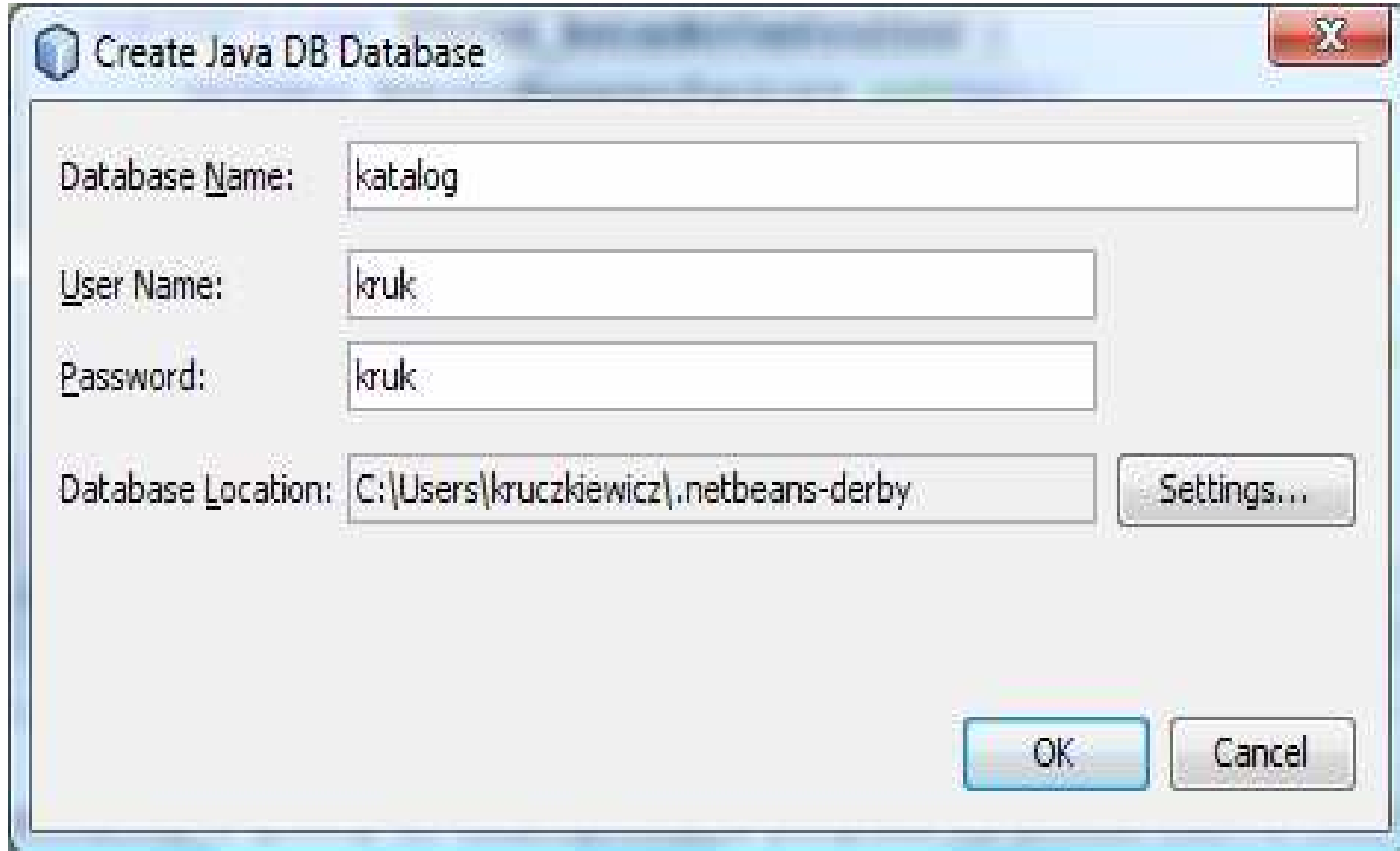
Web Services

Enterprise Beans (2.x)

Servers

```
/*  
 * To change this template, choose Tools  
 * and open the template in the editor.  
 */  
  
package protsym;  
  
import java.util.Random;  
  
/**  
 *  
 * @author kruczkiewicz  
 */  
public class Main {  
    /**  
     * @param args the command line argu
```

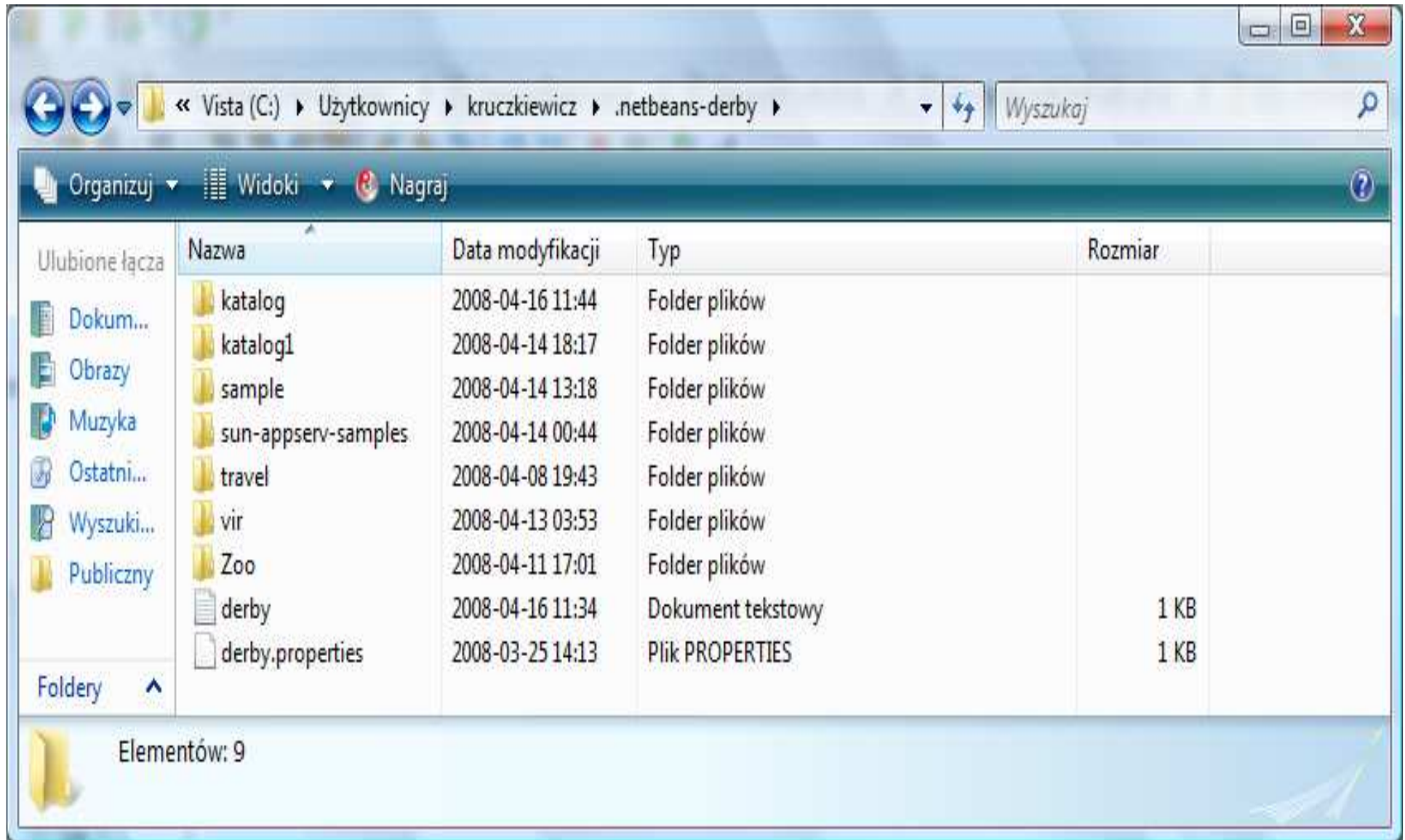
Zakładanie bazy danych **katalog** w systemie baz danych Derby (3)



The screenshot shows a dialog box titled "Create Java DB Database" with a close button (X) in the top right corner. The dialog contains the following fields and buttons:

- Database Name:** katalog
- User Name:** kruk
- Password:** kruk
- Database Location:** C:\Users\kruczkiewicz\.netbeans-derby
- Settings...** button
- OK** button
- Cancel** button

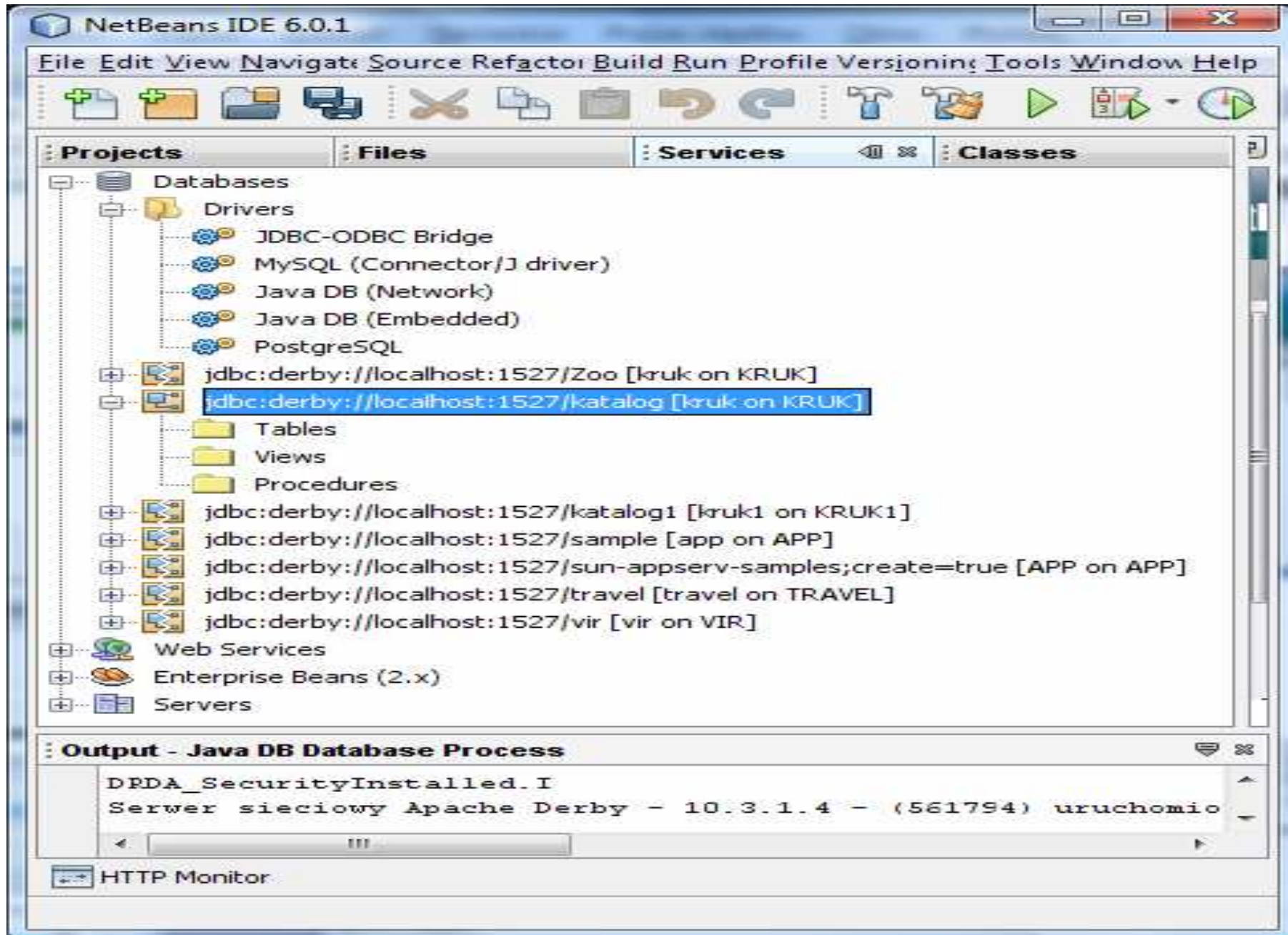
Założono nową bazę danych **katalog** w domyślnym katalogu dla baz danych typu Derby (4)



1.2. Łączenie z pustą bazą danych (5)

The screenshot displays the NetBeans IDE 6.0.1 interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Build, Run, Profile, Versioning, Tools, Window, and Help. The toolbar contains various icons for file operations and development tools. The left sidebar shows the 'Projects' view with a tree structure under 'Databases' containing several Derby connections. A context menu is open over the connection 'jdbc:derby://localhost:1527/katalo', listing options: Connect..., Disconnect, Execute Command..., Refresh, Delete, and Properties. The main workspace shows the 'Welcome to NetBeans IDE 6.0.1' screen with a 'My NetBeans' section. A prominent error message states 'Cannot connect to internet.' with a 'Proxy Configuration...' button. Below this, there are sections for 'Featured Demo', 'ALL NEWS >>', 'ALL ARTICLES >>', and 'ALL BLOGS >>', each with a 'Cannot connect to internet.' message. The bottom status bar shows the 'Output - Java DB Database Process' window with the text: 'DRDA_SecurityInstalled.I', 'Serwer sieciowy Apache Derby - 10.3.1.4 - (561794) uruchomiony i gotowy do zaakceptowania połączeń na porcie 1527 w 2008-04-16 09:34:18.928 GMT |'. The Windows taskbar at the bottom shows the system tray with the time 11:55 and the language set to PL.

Połączenie z pustą bazą danych (6)



2. Tworzenie warstwy integracji w projekcie Java Application. Zastosowanie wzorców projektowych typu **Domain Store i **Transfer Object**.**

Przykład mapowania modelu obiektowego w
środowisku NetBeans 6.01
za pomocą technologii TopLink (przykład z
Laboratorium3_4).

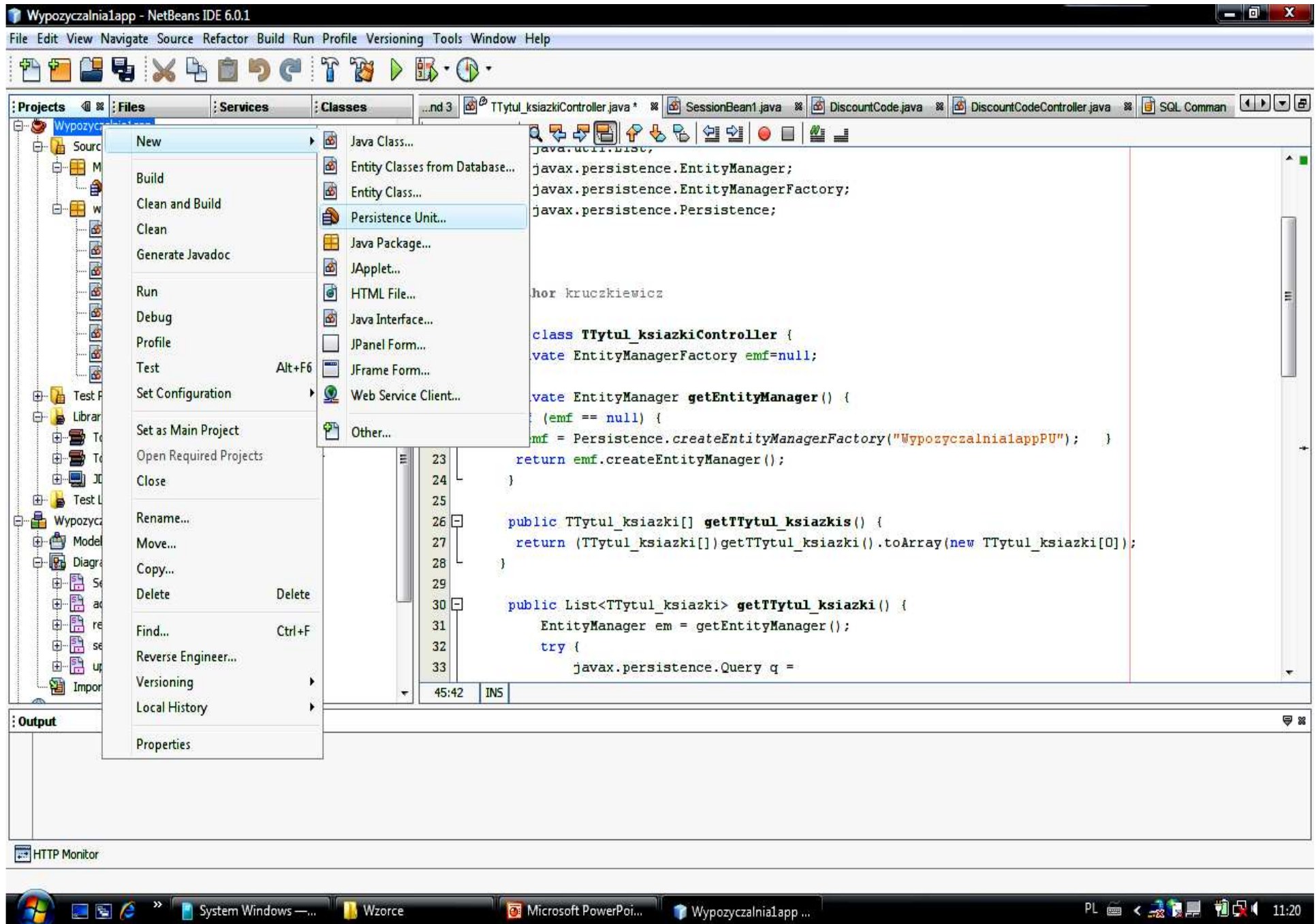
Wprowadzanie warstwy integracji i przekształcanie klas modelu obiektowego na encje (1)

- Należy wstawić moduł typu **Persistence Unit** - prawym klawiszem kliknąć na nazwę projektu w zakładce **Projects**, wybrać **New\ Persistence Unit** (slajd 1). Jeśli w liście nie ma pozycji **Persistence Unit**, należy wybrać pozycję **Other**, wybrać w formularzu **New File** w lewym oknie **Categories** pozycję **Persistence** i w prawym oknie **FileTypes** typ pliku **Persistence Unit**.
- Po naciśnięciu **Next** należy ustalić nazwę modułu. Nazwa modułu wynika z nazwy projektu z przyrostkiem PU. Można zmienić tę nazwę. Należy wybrać **Persistence Library** typu **TopLink** oraz połączyć się z pustą bazą danych założoną w pierwszym etapie prac wybierając właściwy łańcuch połączenia z listy **Data Source**.
- W formularzu tworzenia modułu należy wybrać **Strategy Generation Table** typu **Create** (slajd (3)) – jeśli tabel nie ma, zostaną utworzone, w przeciwnym wypadku nie będą tworzone nowe tabele (sytuacja sygnalizowana wyjątkami). W katalogu projektu **Source Packages – META-INF** pojawia się plik **persitence.xml** i w katalogu **Libraries** pojawiają się biblioteki technologii TopLink w plikach **jar** – zaznaczono je na slajdzie (4)
- Na slajdzie (4) pokazano docelową zawartość projektu typu **Java Application** – należy teraz utworzyć klasy typu **Controller** dla każdej z utrwalanych klas. W przypadku dziedziczenia należy utworzyć te klasy dla klas bazowych. Klasy typu **Controller** są fasadami warstwy integracji dla każdej z utrwalanych klas.
- W celu utworzenia każdego z plików **TTytul_ksiazkiController.java** oraz **TEgzemplarzController.java**, należy prawym klawiszem kliknąć na nazwę projektu w zakładce **Projects**, wybrać **New\Java Class**. Jeśli w liście nie ma pozycji **Java Class**, należy wybrać pozycję **Other**, wybrać w formularzu **New File** w lewym oknie **Categories** pozycję **Java** i w prawym oknie **FileTypes** typ pliku **Java Class**. Po naciśnięciu **Next** należy podać nazwę pliku w pozycji **Class Name (TTytul_ksiazkiController)** i wybrać z listy **Package** pakiet, w którym znajdują się klasy warstwy biznesowej (TTytul_ksiazki.java itd..) np. wypożyczalnia1app. Nacisnąć **Finish**. Całą procedurę powtórzyć do utworzenia pliku **TEgzemplarzController.java**. Po zakończeniu tych czynności powstają pliki zawierające klasy typu **public** z pustym ciałem.

Wprowadzanie warstwy integracji i przekształcanie klas modelu obiektowego na encje (2)

- Na slajdach 5-13 pokazano przebieg tworzenia kodu dla klasy typu **Controller** dla **TTytuł_książki**. Należy napisać taki kod zgodnie ze wskazówkami również dla klasy **TEgzemplarz**. Podczas pisania metody `getEntityManager()` łańcuch w liście parametrów metody `createEntityManagerFactory` powinien być nazwą modułu `Persistence Unit`. Oba pliki mogą znajdować się w katalogu klas modelu biznesowego lub nowym pakiecie tego projektu. W celu ułatwienia czynności po zakończeniu tworzenia ciała klasy **TTytuł_książkiController** można skopiować je i wkleić do ciała klasy **TEgzemplarzController**, zamienić ciąg znaków **TTytuł_książki** na **TEgzemplarz** za pomocą **CTRL+H**. Podczas tworzenia kodu należy uzupełniać import pakietów za pomocą opcji **Fix Imports** (prawym klawiszem myszy należy klikać na ciało klasy i wybrać z wyskakującego menu **Fix Imports** i wybierać klasy z pakietu **javax.persistence**. Dla typu **List** wybrać pakiet **java.util**). *Po tych czynnościach jedynie linie z wywoływana metodą `find` są niekompilowane. Sytuacja ta zmieni się po przekształceniu klas modelu obiektowego na rodzaj **Entity**.*
- Na slajdach 14 do 15 (wyłączając 15) podano wskazówki dotyczące zamiany klas **TTytuł_książki**, **TTytuł_książki_na_kasce**, **TEgzemplarz**, **TEgzemplarz_termin** na klasy z adnotacjami `@Entity`. W klasie **TEgzemplarz** wprowadzono wirtualne metody: **public Date getTermin()** i **public void setTermin(Date termin)**. Podczas tworzenia kodu należy uzupełniać import pakietów za pomocą opcji **Fix Imports** (prawym klawiszem myszy należy klikać na ciało klasy i wybrać z wyskakującego menu **Fix Imports** i wybierać klasy z pakietu **javax.persistence**).
- Zgodnie ze slajdem 15, należy w zakładce **Projects** wybrać w projekcie zakładkę **META-INF** i otworzyć plik **persistence.xml** w trybie **Design**. W formularzu pliku XML należy za pomocą klawisza **Add class** wstawić z listy utworzone cztery klasy typu **Entity**.
- Należy skompilować uzupełniany projekt – tutaj Wypożyczalnia1app

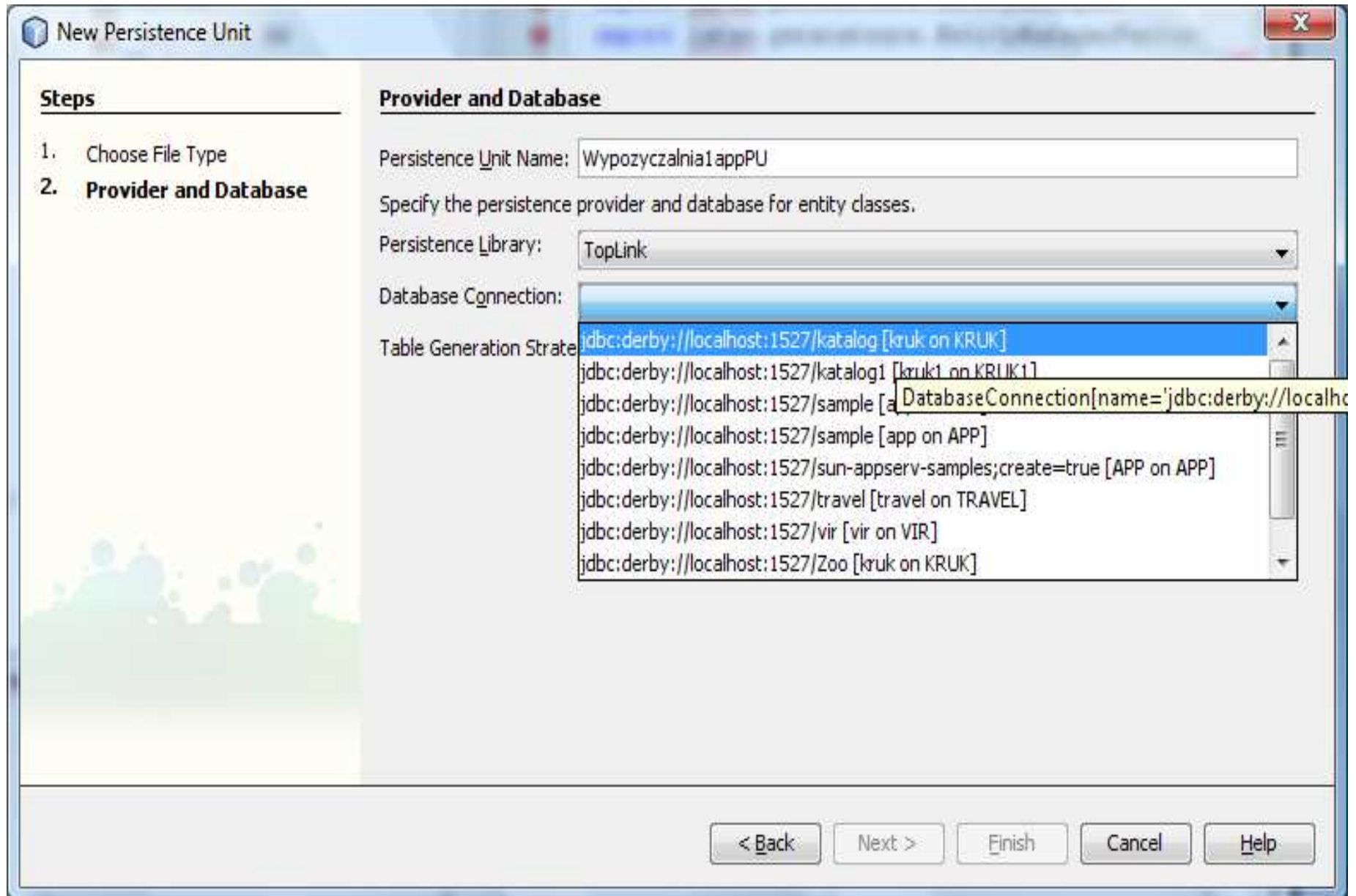
2.1. Wstawianie do projektu typu **Java Application** modułu typu **Persistence Unit (1)**



The screenshot displays the NetBeans IDE 6.0.1 interface. The 'New' menu is open, and 'Persistence Unit...' is selected. The background shows the source code of a Java class named `Ttytul_książkiController.java`. The code includes imports for `java.util.List`, `javax.persistence.EntityManager`, `javax.persistence.EntityManagerFactory`, and `javax.persistence.Persistence`. The class `Ttytul_książkiController` has a private field `EntityManagerFactory emf` and methods `getEntityManager()`, `getTtytul_książkis()`, and `getTtytul_książki()`. The `getTtytul_książki()` method uses `EntityManager` and `Query` to retrieve data.

```
java.util.List,  
javax.persistence.EntityManager;  
javax.persistence.EntityManagerFactory;  
javax.persistence.Persistence;  
  
hor kruczkiewicz  
  
class Ttytul_książkiController {  
    private EntityManagerFactory emf=null;  
  
    private EntityManager getEntityManager() {  
        if (emf == null) {  
            emf = Persistence.createEntityManagerFactory("Wypożyczalnia1appPU");  
        }  
        return emf.createEntityManager();  
    }  
  
    public Ttytul_książki[] getTtytul_książkis() {  
        return (Ttytul_książki[])getTtytul_książki().toArray(new Ttytul_książki[0]);  
    }  
  
    public List<Ttytul_książki> getTtytul_książki() {  
        EntityManager em = getEntityManager();  
        try {  
            javax.persistence.Query q =
```

Wybór bazy danych, w której będą utrwalane obiekty – pustej (2)



Tworzenie modułu utrwalania danych dla technologii TopLink (3)

New Persistence Unit

Steps

1. Choose File Type
2. **Provider and Database**

Provider and Database

Persistence Unit Name: Wypożyczalnia1appPU

Specify the persistence provider and database for entity classes.

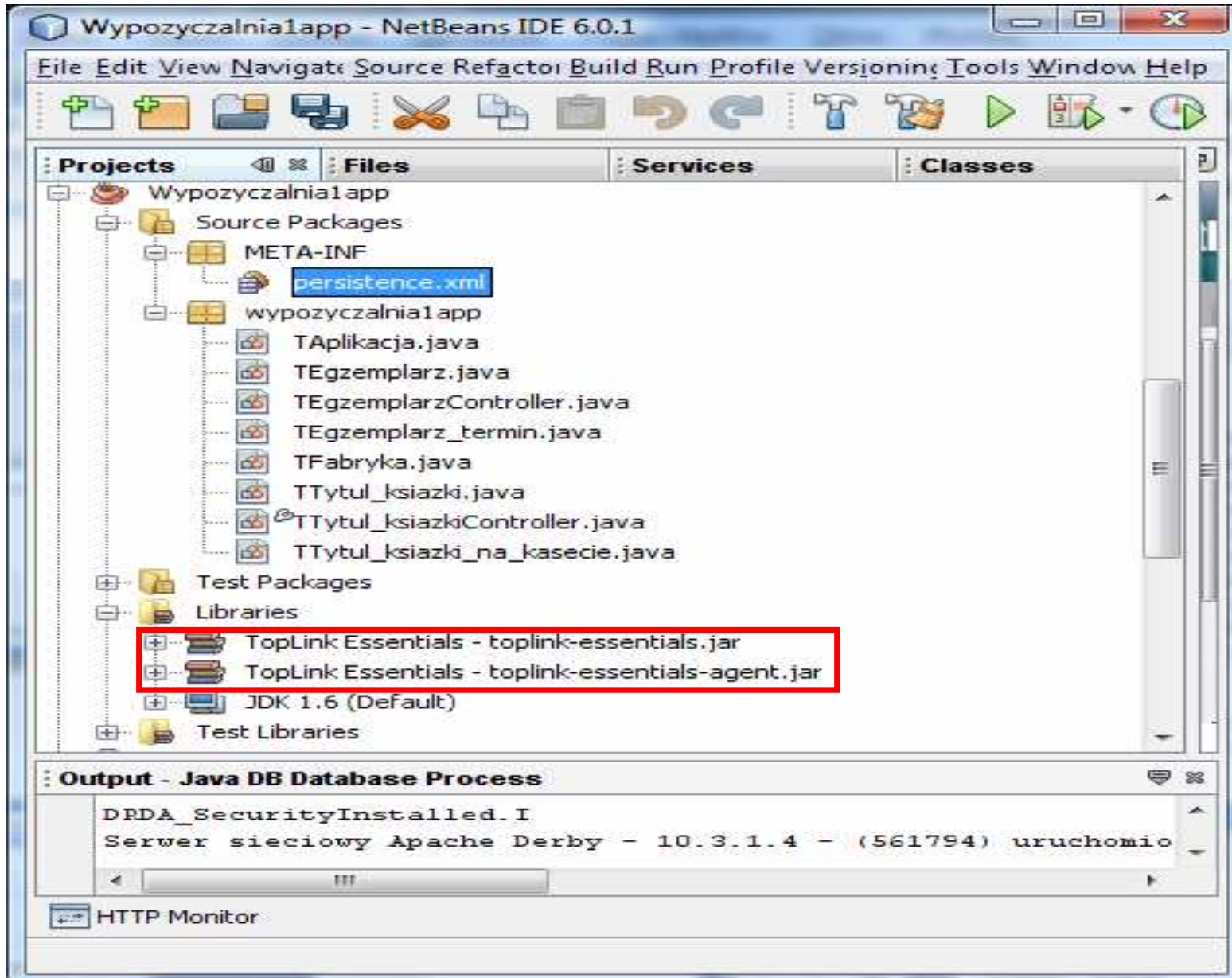
Persistence Library: TopLink

Database Connection: jdbc:derby://localhost:1527/katalog [kruk on KRUK]

Table Generation Strategy: Create Drop and Create None

< Back Next > **Finish** Cancel Help

Plik **persistence.xml** reprezentujący moduł utrwalania danych typu **TopLink**
Uzupełnianie zawartości projektu typu **Java Application** o klasy typu **Controller** dla każdej z utrwalanych klas modelu obiektowego (4)



2.2. Inżynieria odwrotna dla utworzonych przez programistę klas typu **Controller** dla każdej z utrwalanych klas modelu obiektowego (**TTytul_książki**) (5)

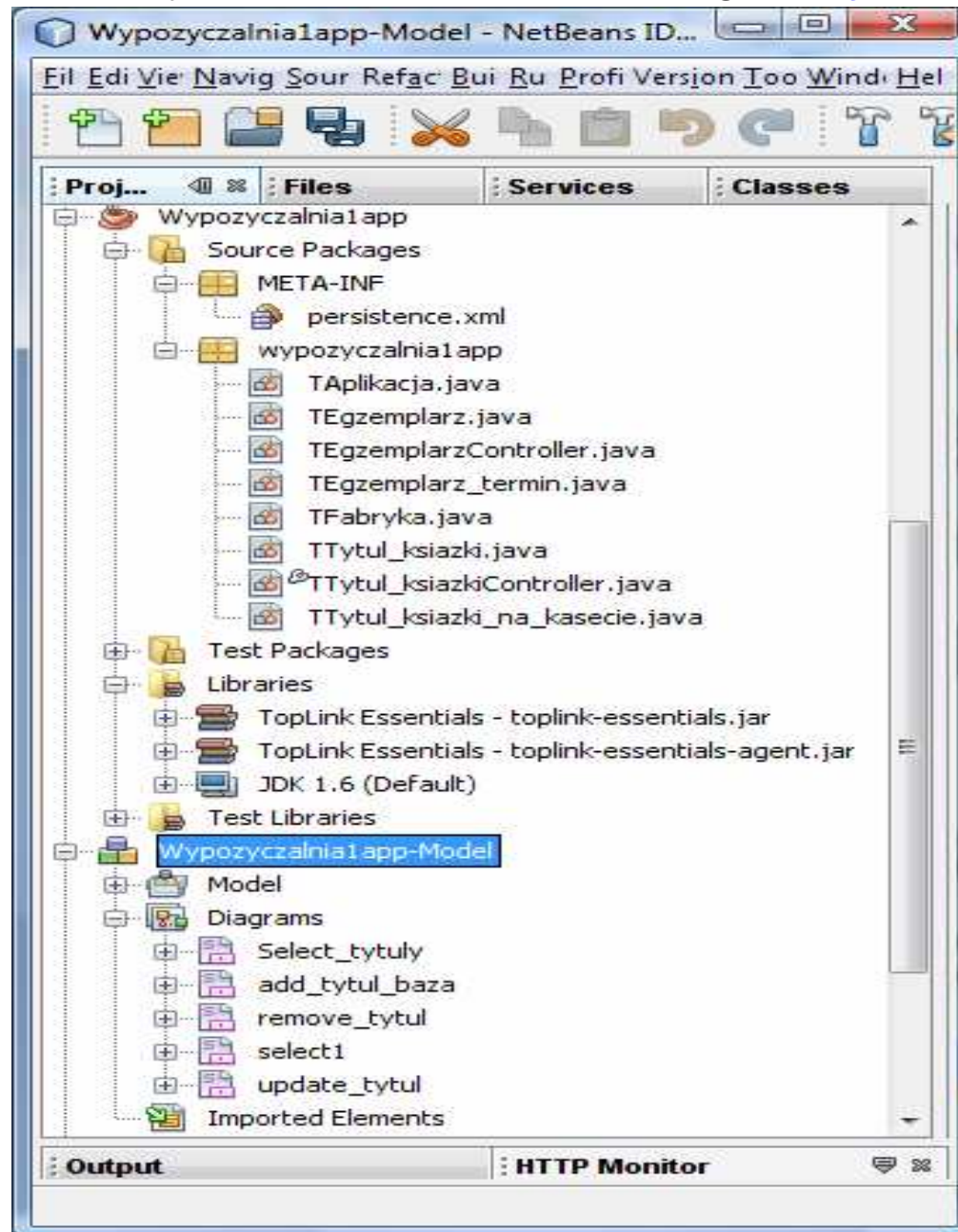
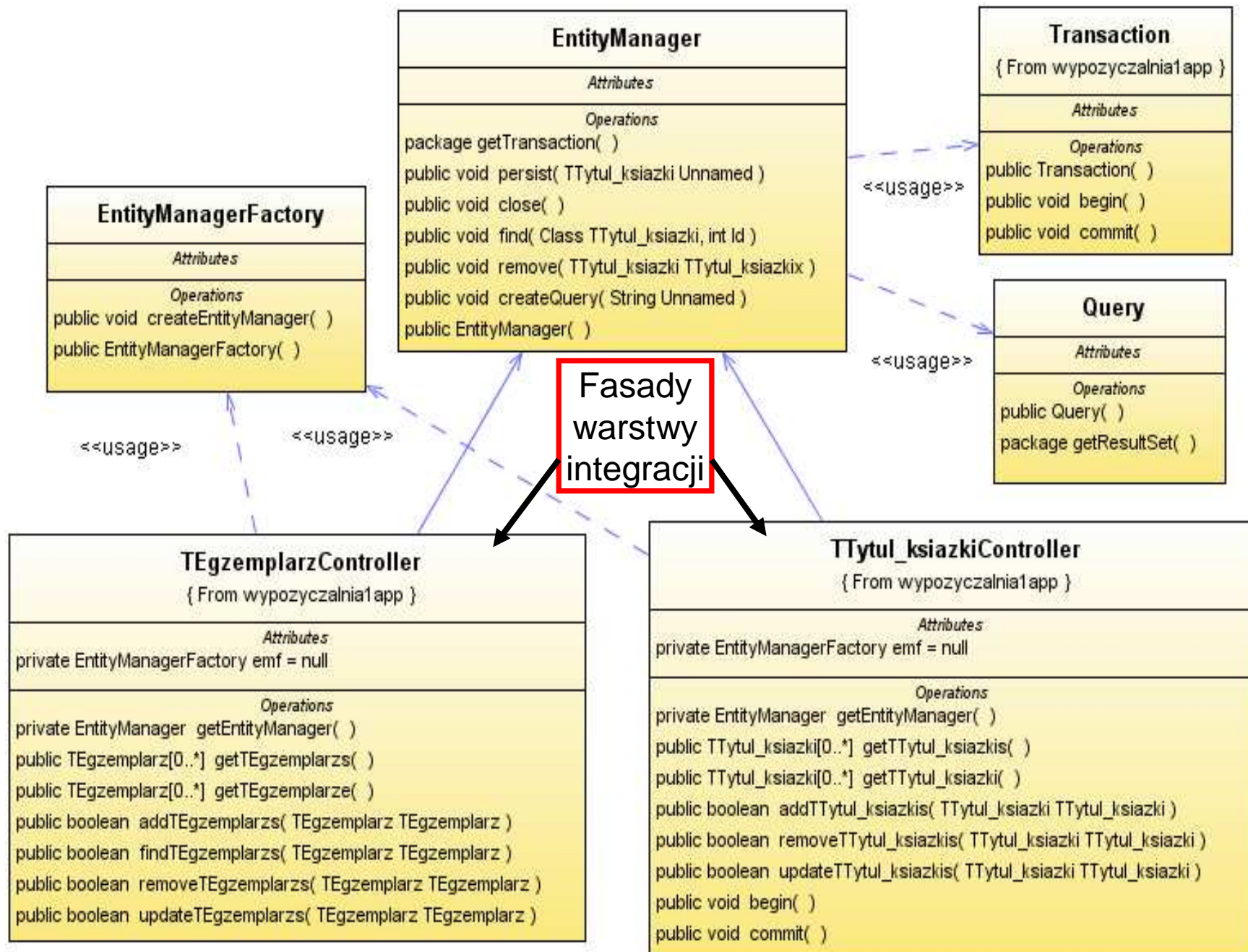
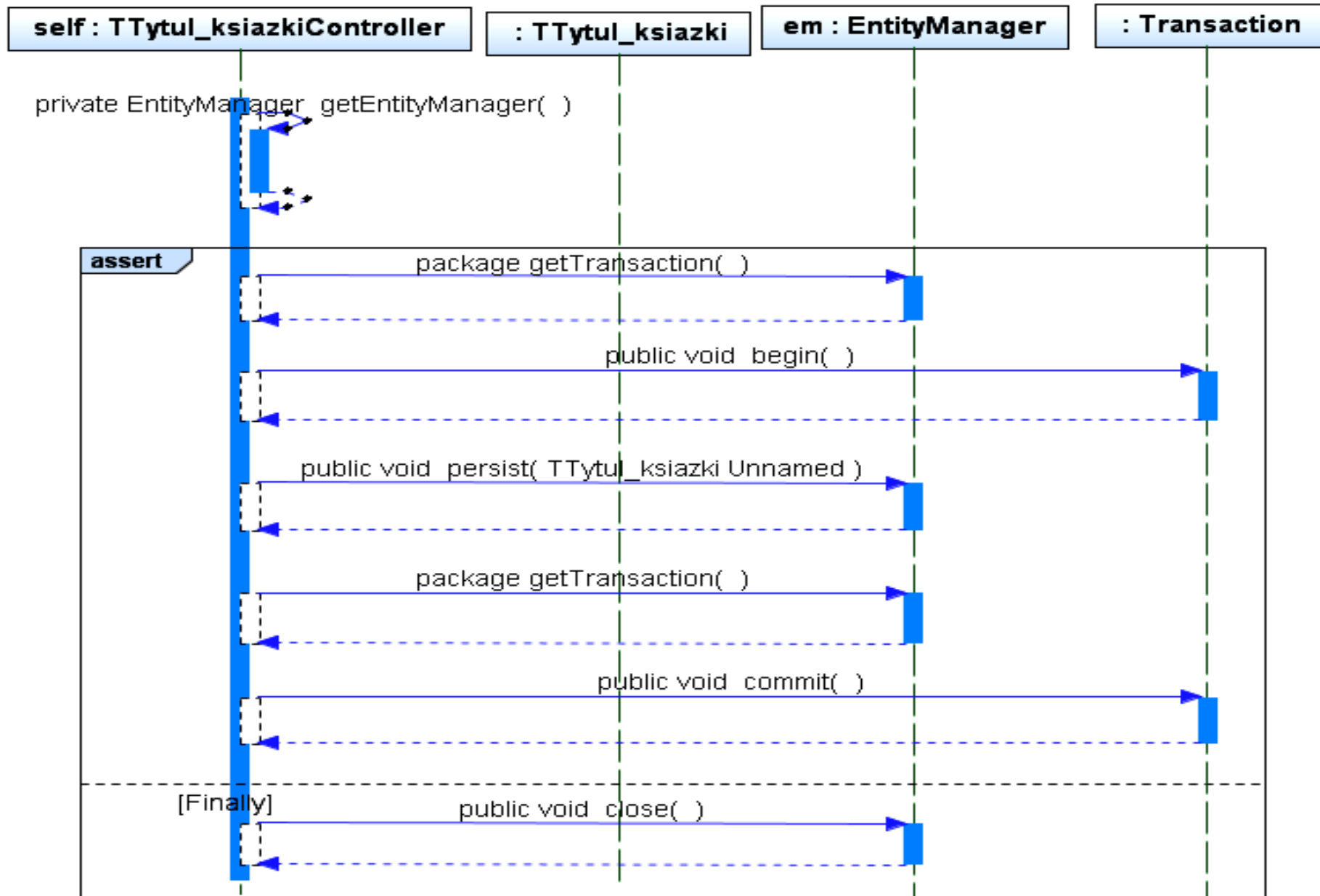


Diagram klas – uproszczony schemat warstwy integracji (6)

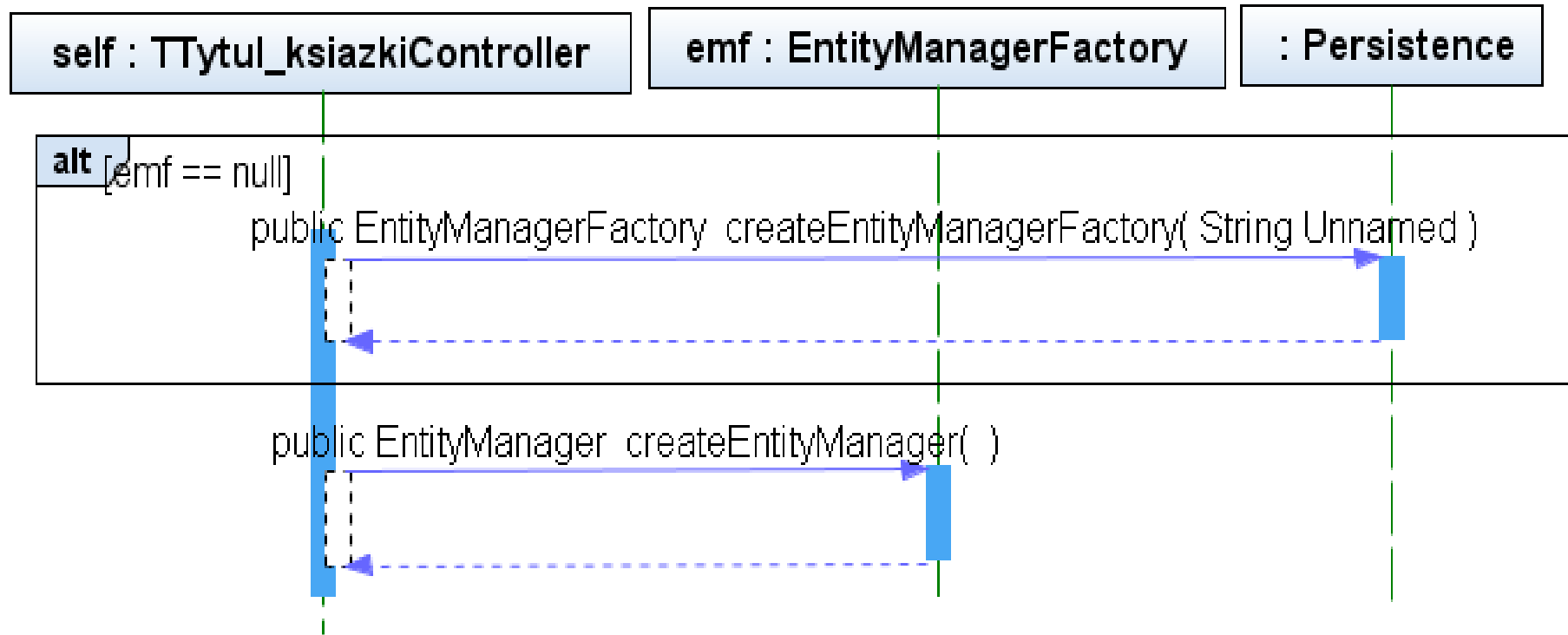


Dodawanie tytułów do bazy danych -

public boolean addTTytul_ksiazkis(TTytul_ksiazki TTytul_ksiazki) (7)



Dodawanie tytułów do bazy danych -
private EntityManager getEntityManager() (8)



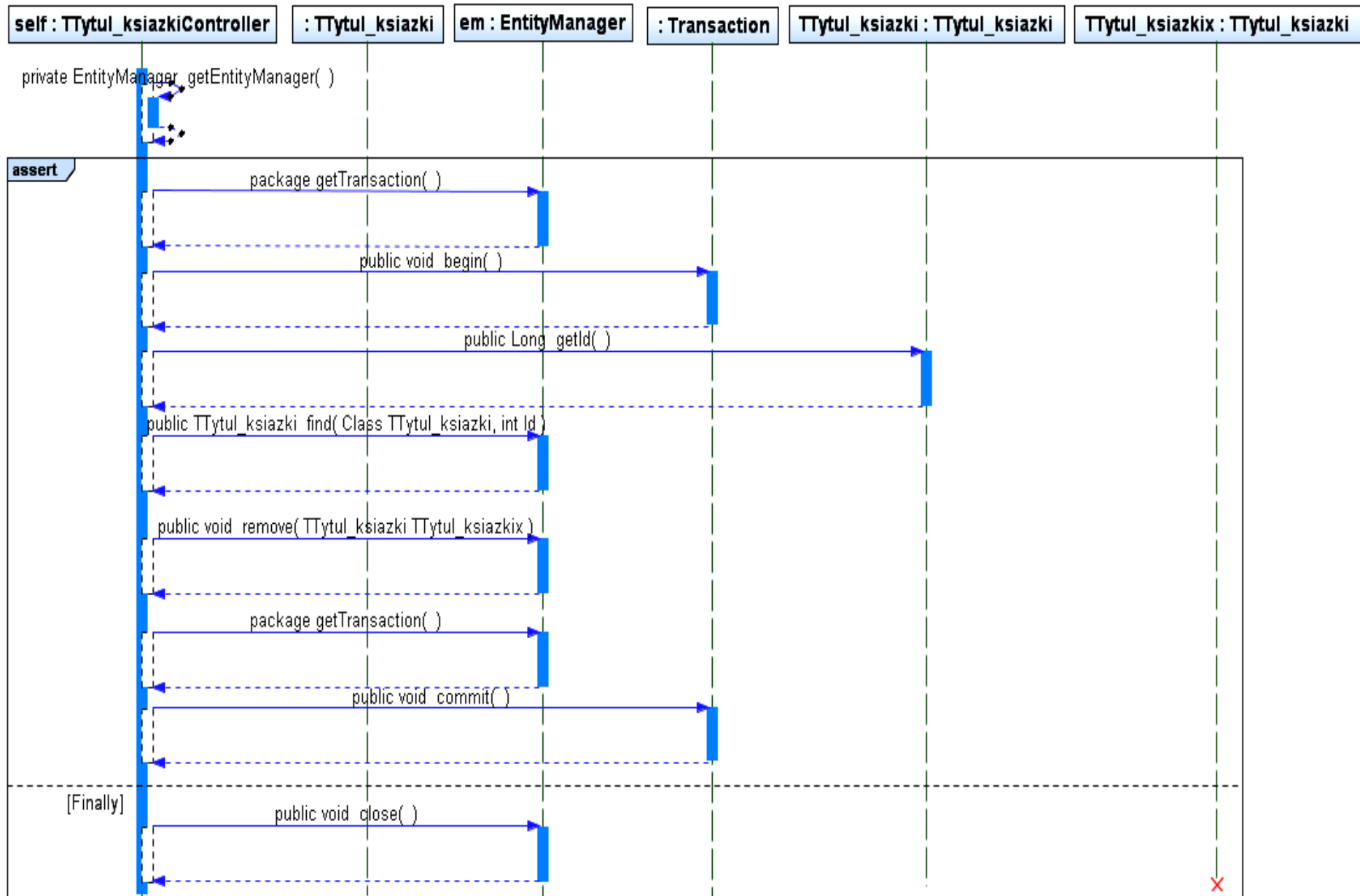
Klasa typu Controller dla każdej z klas utrwalanych obiektów – (9)
Realizacja wzorców typu [Domain Store](#) i [Transfer Object](#)

```
public class TTytul_ksiazkiController {  
    private EntityManagerFactory emf=null;  
  
    private EntityManager getEntityManager() {  
        if (emf == null) {  
            emf = Persistence.createEntityManagerFactory(  
                "Wypożyczalnia1appPU");  
        }  
        return emf.createEntityManager();  
    }  
}
```

```
public boolean addTTytul_ksiazkis(  
    TTYtul_ksiazki TTYtul_ksiazki) {  
    EntityManager em = getEntityManager();  
try {  
        em.getTransaction().begin();  
        em.persist(TTYtul_ksiazki);  
        em.getTransaction().commit();  
    } finally {  
        em.close();  
        return false; }  
}
```


Usuwanie tytułów z bazy danych (10)

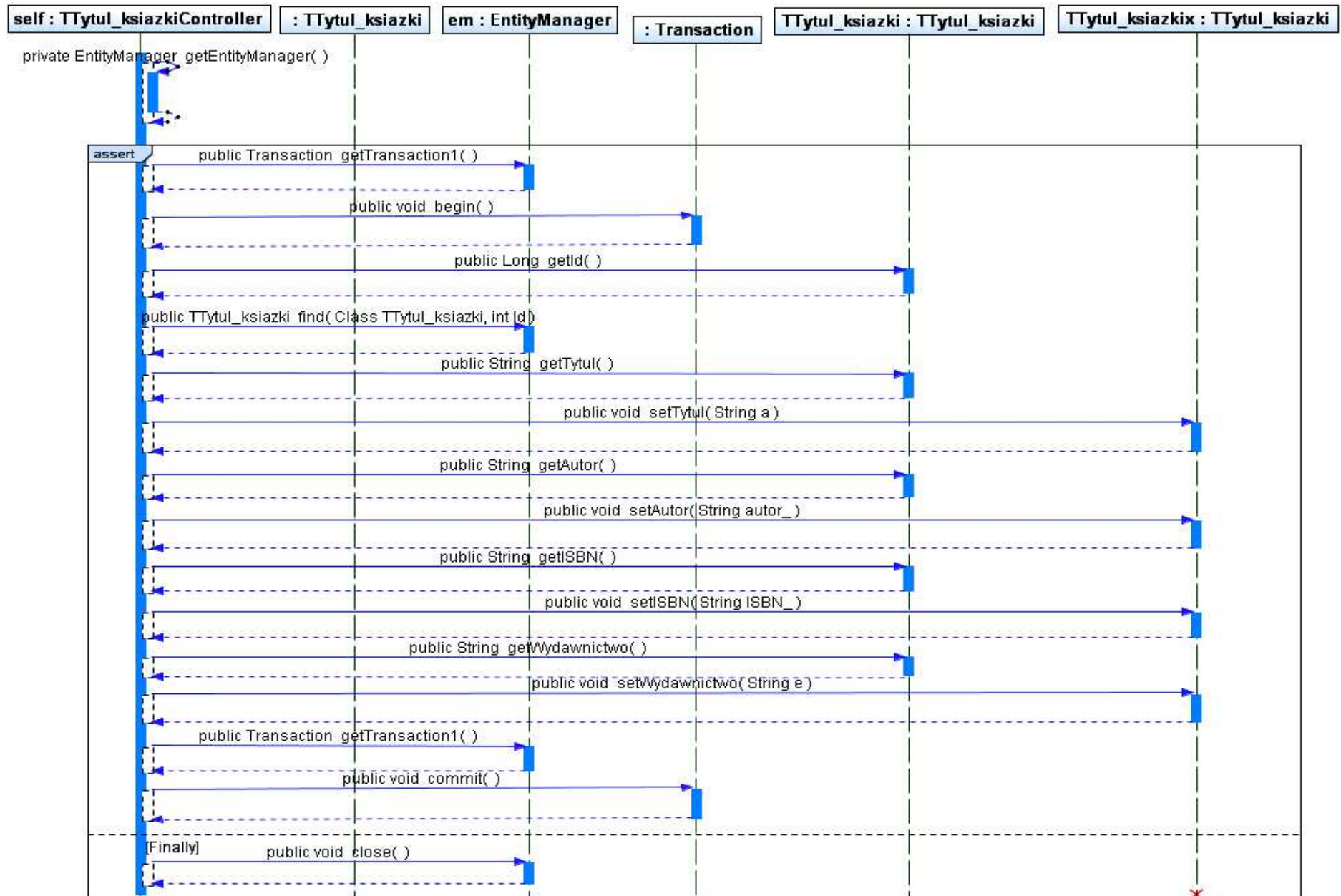
public boolean removeTTytul_ksiazkis(TTytul_ksiazki TTytul_ksiazki)



```
public boolean removeTTytul_ksiazkis(
    TTYtul_ksiazki TTYtul_ksiazki) {
    EntityManager em = getEntityManager();
try {
        em.getTransaction().begin();
        TTYtul_ksiazki TTYtul_ksiazkix =
            em.find(TTYtul_ksiazki.class,
                TTYtul_ksiazki.getId());
        em.remove(TTYtul_ksiazkix);
        em.getTransaction().commit();
    } finally {
        em.close();
        return false; }
}
```

Modyfikacja danych w bazie danych (11)

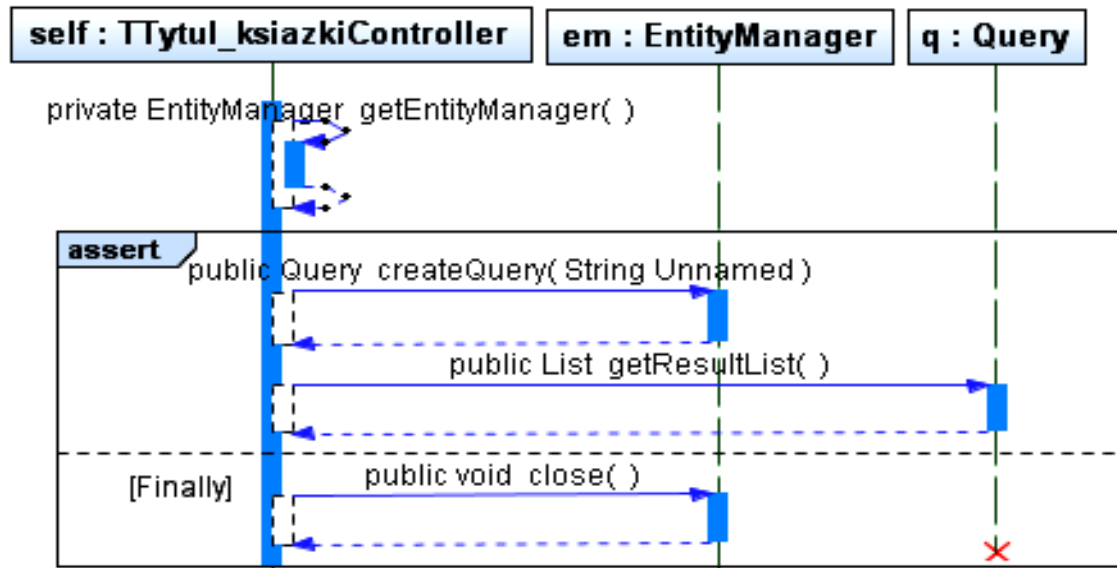
public boolean updateTTytul_ksiazkis(TTytul_ksiazki TTytul_ksiazki)



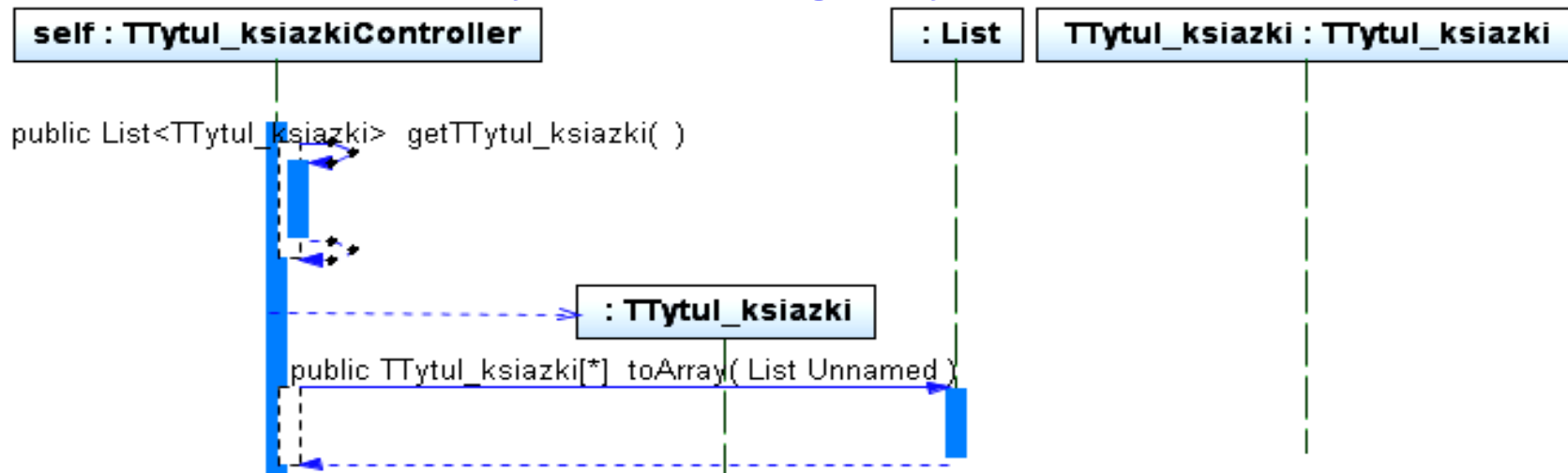
```
public boolean updateTTytul_ksiazkis(TTytul_ksiazki TTytul_ksiazki)
{ EntityManager em = getEntityManager();
  try
  {
    em.getTransaction().begin();
    TTytul_ksiazki TTytul_ksiazkix =
      em.find(TTytul_ksiazki.class, TTytul_ksiazki.getId());
    TTytul_ksiazkix.setTytul(TTytul_ksiazki.getTytul());
    TTytul_ksiazkix.setAutor(TTytul_ksiazki.getAutor());
    TTytul_ksiazkix.setISBN(TTytul_ksiazki.getISBN());
    TTytul_ksiazkix.setWydawnictwo(TTytul_ksiazki.getWydawnictwo());
    em.getTransaction().commit();
  }
  finally
  {
    em.close();
    return false; }
}
```

Odczytanie tytułów z bazy danych w postaci listy i tablicy TTytul_książki (12) (wzorzec typu **Transfer Object** – TTytul_książki[0..*])

public List<TTytul_książki> getTTytul_książki()



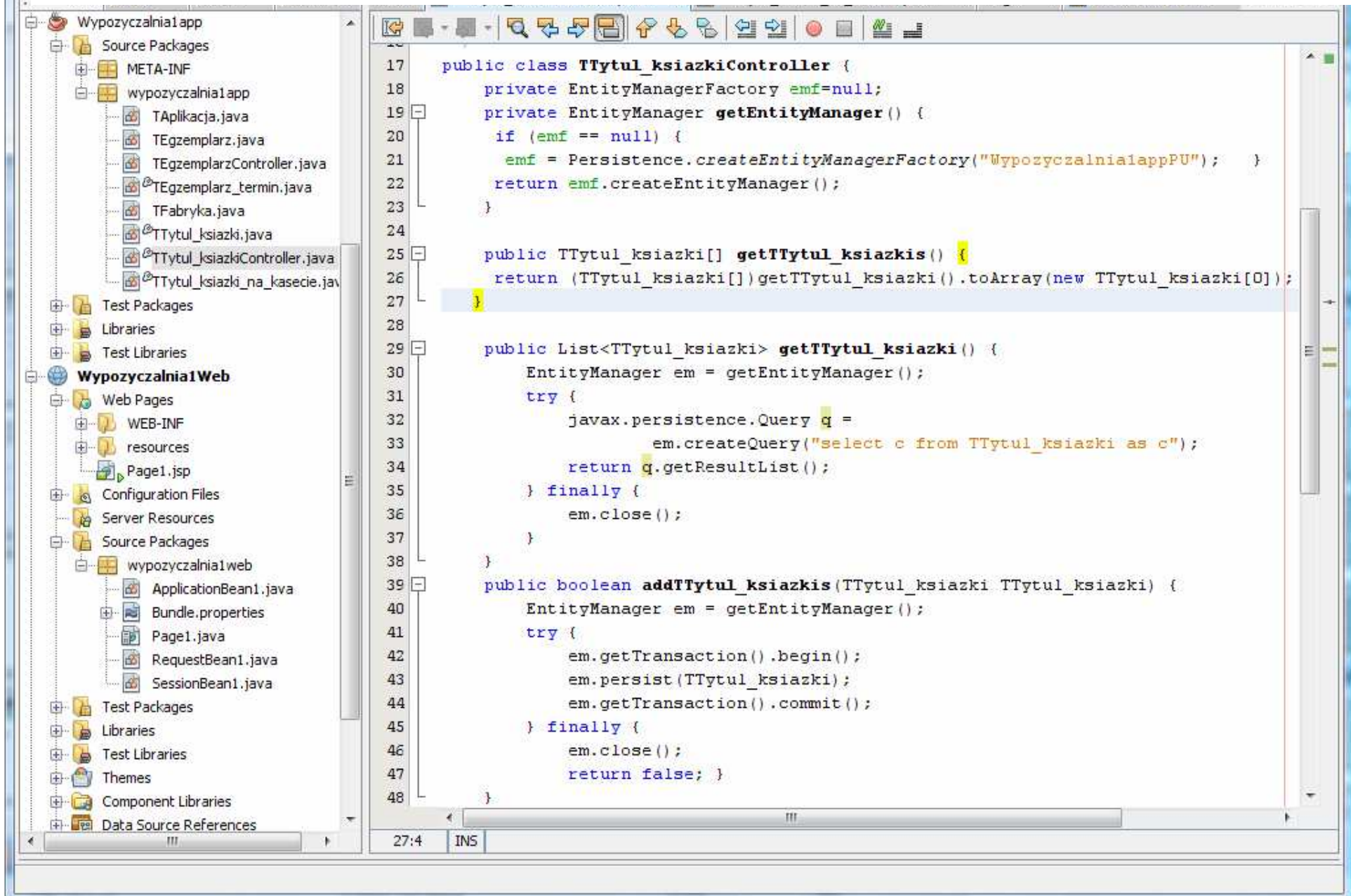
public TTytul_książki[] getTTytul_książkis()



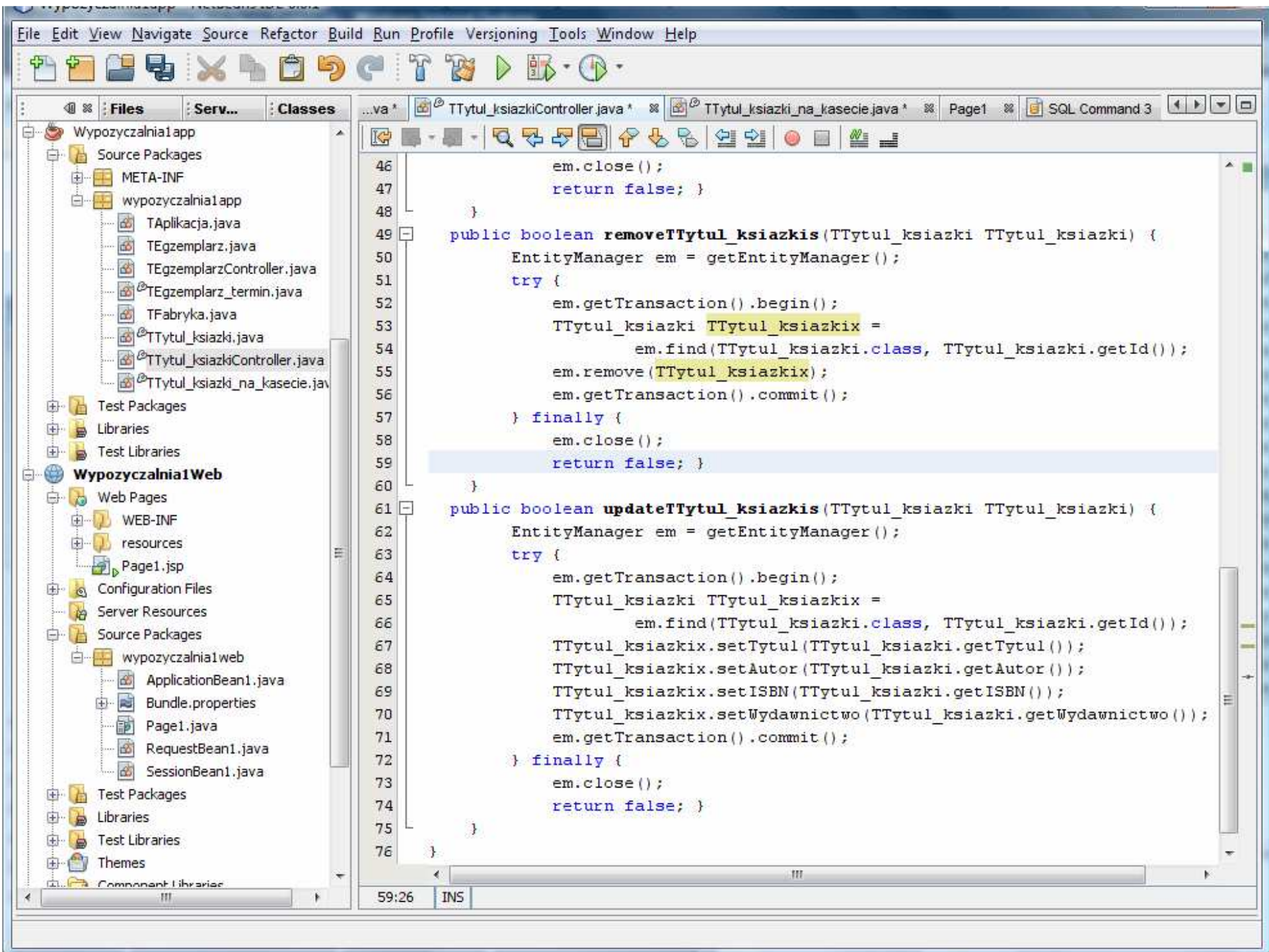
```
public TTytul_ksiazki[] getTTytul_ksiazkis()
{
    return (TTytul_ksiazki[]) getTTytul_ksiazki().toArray(
                                                new TTytul_ksiazki[0]);
}
```

```
public List<TTytul_ksiazki> getTTytul_ksiazki()
{
    EntityManager em = getEntityManager();
    try {
        javax.persistence.Query q =
            em.createQuery("select c from TTytul_ksiazki as c");
        return q.getResultList();
    } finally {
        em.close();
    }
}
```

Klasa warstwy integracji (szablon) – dla każdej klasy typu **Entity** (tutaj dla **TTytul_książki**) (13)



```
17 public class TTytul_książkiController {
18     private EntityManagerFactory emf=null;
19     private EntityManager getEntityManager() {
20         if (emf == null) {
21             emf = Persistence.createEntityManagerFactory("Wypożyczalnia1appPU"); }
22     return emf.createEntityManager();
23 }
24
25 public TTytul_książki[] getTTytul_książkis() {
26     return (TTytul_książki[])getTTytul_książki().toArray(new TTytul_książki[0]);
27 }
28
29 public List<TTytul_książki> getTTytul_książki() {
30     EntityManager em = getEntityManager();
31     try {
32         javax.persistence.Query q =
33             em.createQuery("select c from TTytul_książki as c");
34         return q.getResultList();
35     } finally {
36         em.close();
37     }
38 }
39 public boolean addTTytul_książkis(TTytul_książki TTytul_książki) {
40     EntityManager em = getEntityManager();
41     try {
42         em.getTransaction().begin();
43         em.persist(TTytul_książki);
44         em.getTransaction().commit();
45     } finally {
46         em.close();
47         return false; }
48 }
```

(14a) – Standaryzacja nazw metod dostępu do składowych w klasach @Entity - przykłady

1. Należy przyjąć zasadę, że **metody dostępu** do składowych typu **set** i **get** przyjmują pełne nazwy przez dodanie przyrostka jako nazwy atrybutu z zamienioną pierwszą literą na dużą

np. definicja atrybutu – TTytul_książki mTytul_książki;

metoda typu set – **public void** setMTytul_książki (TEgzemplarz val)

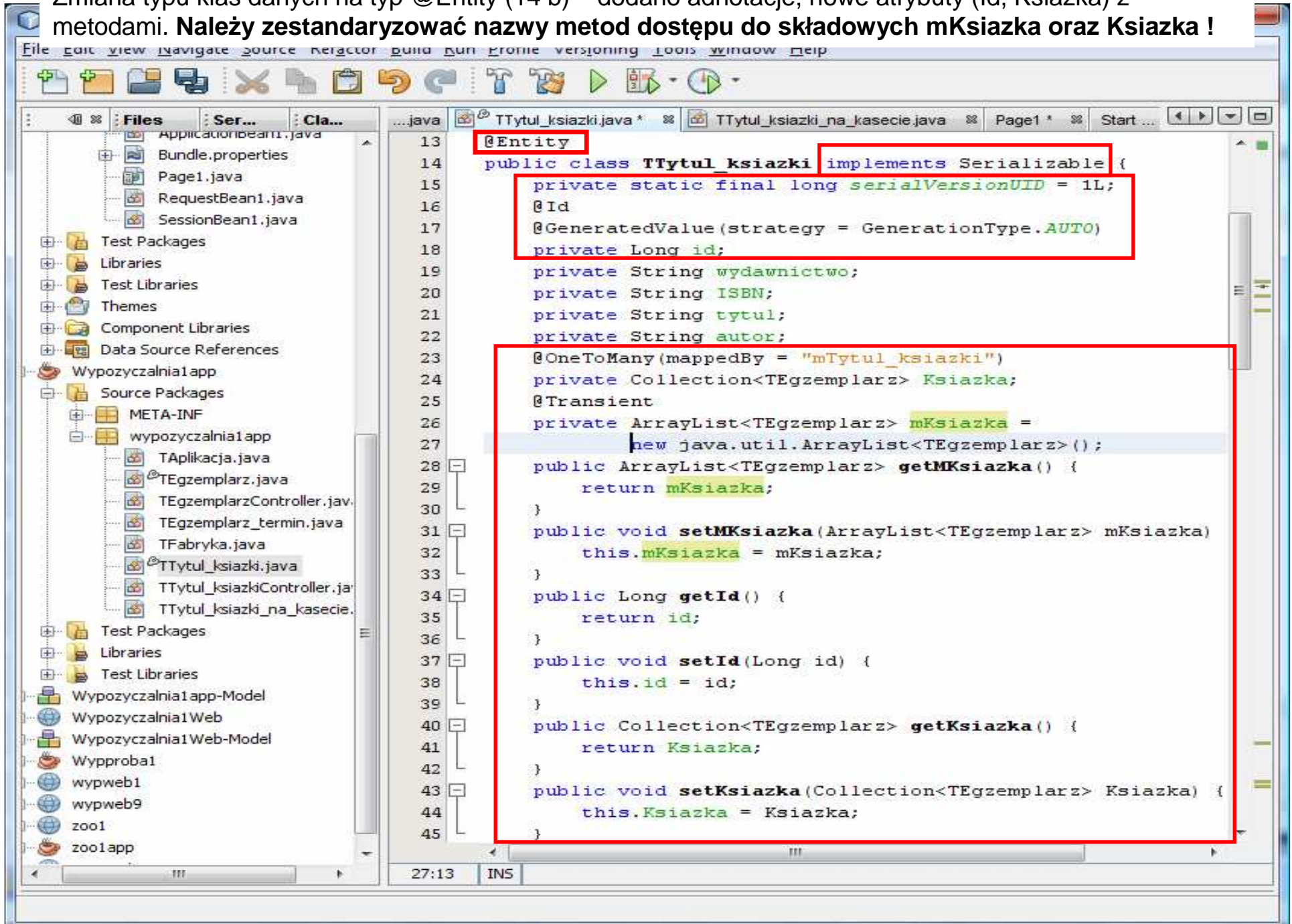
{ mTytul_książki = val; }

metoda typu get - **public** TTytul_książki getMKsiążka ()

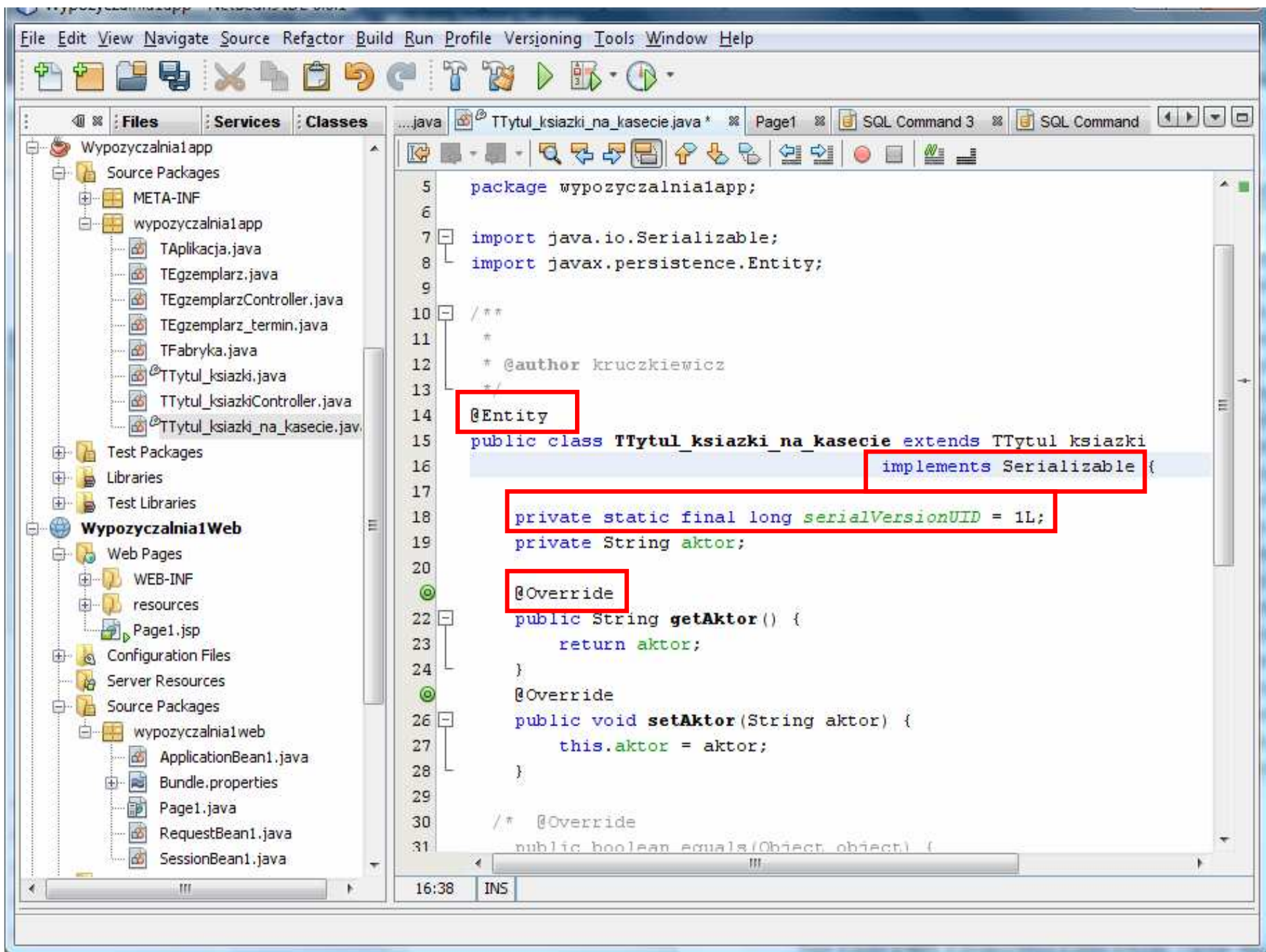
{ **return** mTytul_książki; }

2. Usunąć wszystkie niestandardowe nazwy metod dostępu do każdej ze składowych
3. Wykonać metody typu **set** i **get** zgodnie z p.1 w sposób automatyczny:
 - 3.1. Kliknąć prawym klawiszem myszy na ciało klasy w trybie Java, wybrać **Refactor**, następnie **Encapsulate Fields**
 - 3.2. Zaznaczyć **Create Getter** oraz **Create Setter** i nacisnąć **Refactor**

Zmiana typu klas danych na typ @Entity (14 b) – dodano adnotacje, nowe atrybuty (Id, Ksiazka) z metodami. **Należy zestandaryzować nazwy metod dostępu do składowych mKsiazka oraz Ksiazka !**



```
13 @Entity
14 public class TTytul_ksiazki implements Serializable {
15     private static final long serialVersionUID = 1L;
16     @Id
17     @GeneratedValue(strategy = GenerationType.AUTO)
18     private Long id;
19     private String wydawnictwo;
20     private String ISBN;
21     private String tytul;
22     private String autor;
23     @OneToMany(mappedBy = "mTytul_ksiazki")
24     private Collection<TEgzemplarz> Ksiazka;
25     @Transient
26     private ArrayList<TEgzemplarz> mKsiazka =
27         new java.util.ArrayList<TEgzemplarz>();
28     public ArrayList<TEgzemplarz> getMKsiazka() {
29         return mKsiazka;
30     }
31     public void setMKsiazka(ArrayList<TEgzemplarz> mKsiazka)
32     {
33         this.mKsiazka = mKsiazka;
34     }
35     public Long getId() {
36         return id;
37     }
38     public void setId(Long id) {
39         this.id = id;
40     }
41     public Collection<TEgzemplarz> getKsiazka() {
42         return Ksiazka;
43     }
44     public void setKsiazka(Collection<TEgzemplarz> Ksiazka) {
45         this.Ksiazka = Ksiazka;
46     }
47 }
```

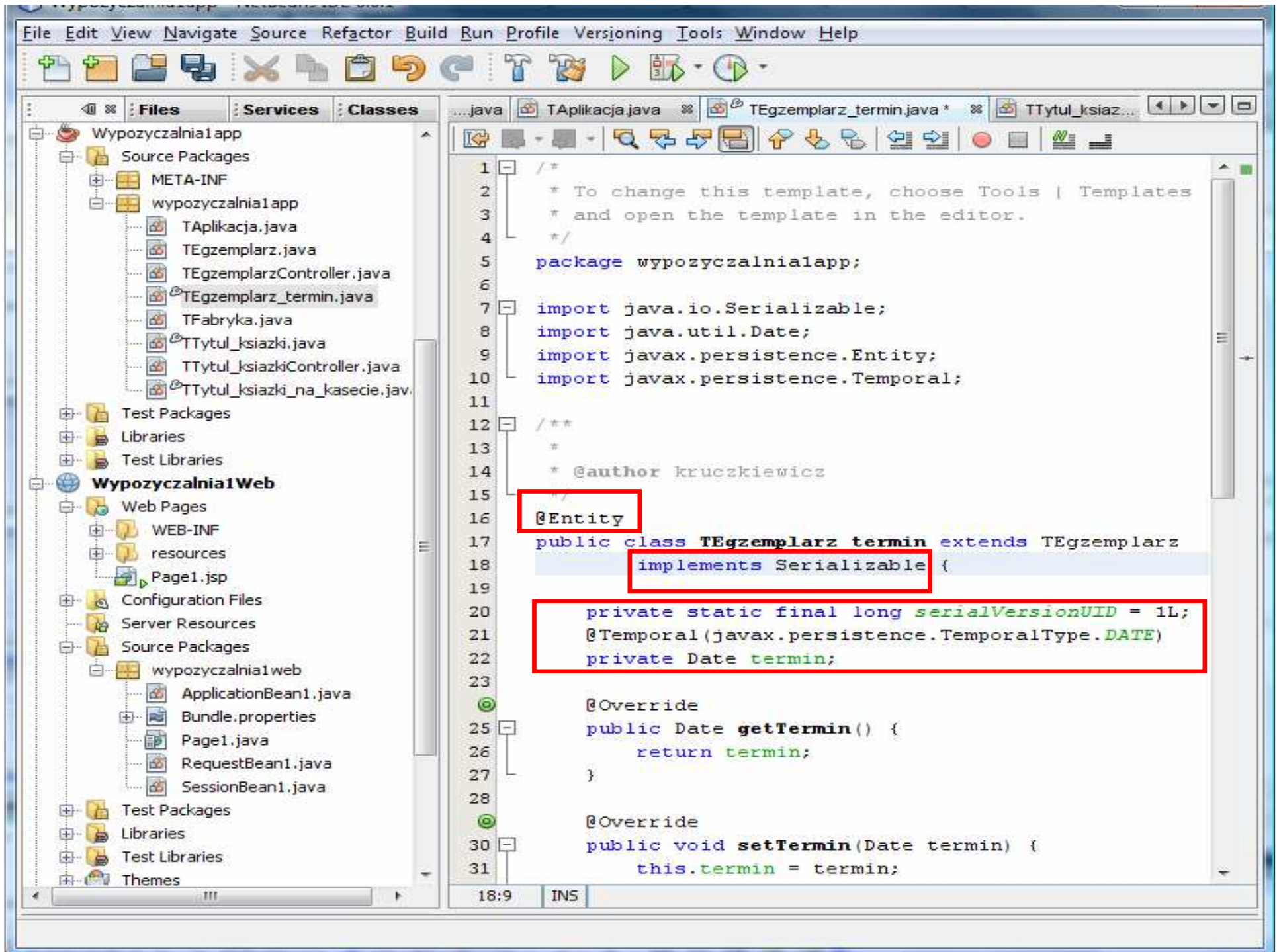



Zmiana typu klas danych na typ @Entity (14 c) – dodano adnotacje, nowy atrybut (Id) z metodami. **Należy zestandaryzować nazwy metod dostępu do składowej mTytul_ksiazki!**

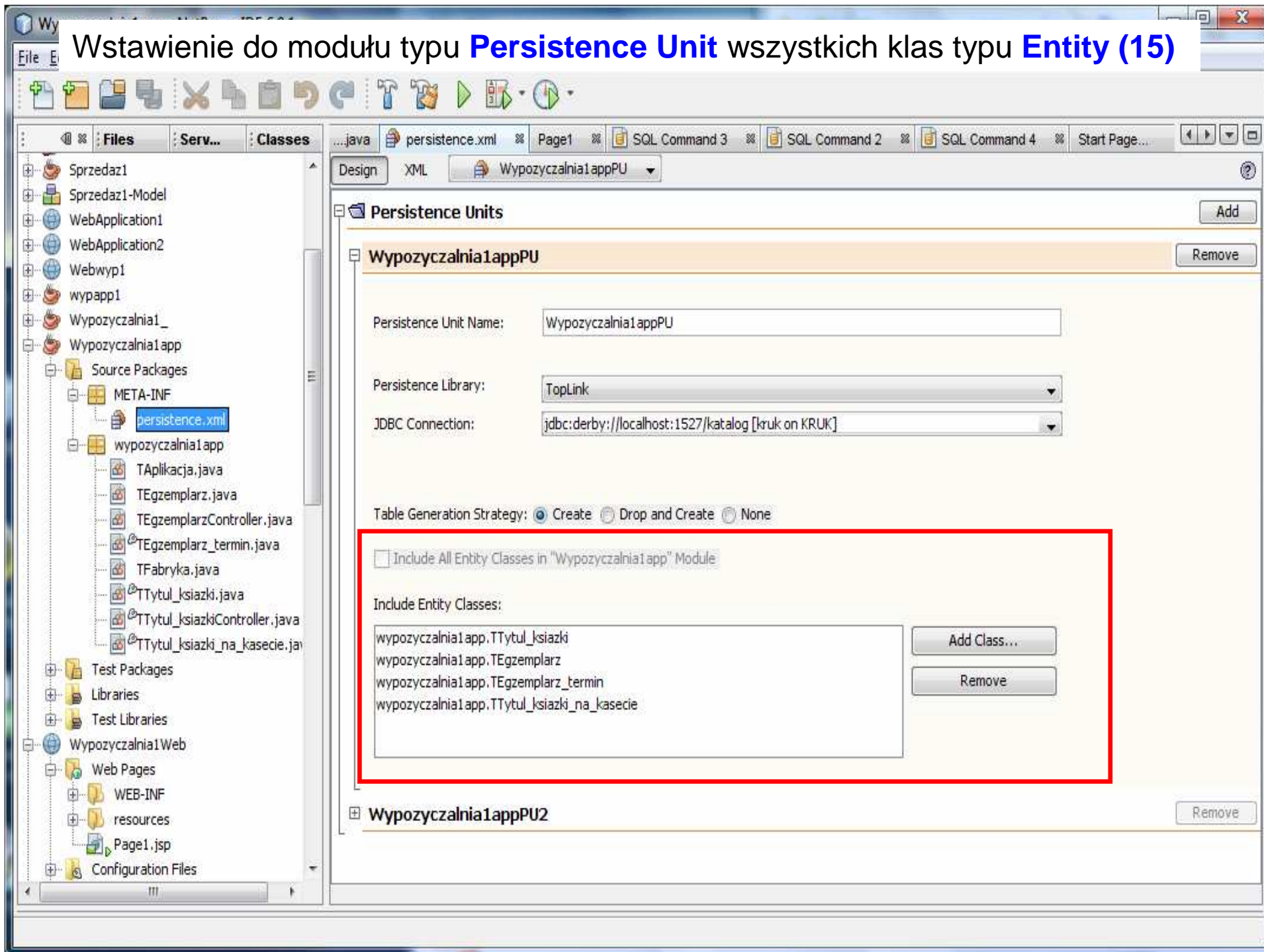
```
11 @Entity
12 public class TEgzemplarz implements Serializable {
13     private static final long serialVersionUID = 1L;
14     @Id
15     @GeneratedValue(strategy = GenerationType.AUTO)
16     private Long id;
17     private int numer;
18     @ManyToOne
19     private TTytul_ksiazki mTytul_ksiazki;
20
21     public TEgzemplarz() {
22     }
23     public Long getId() {
24         return id;
25     }
26     public void setId(Long id) {
27         this.id = id;
28     }
29     public TTytul_ksiazki getMTytul_ksiazki() {
30         return mTytul_ksiazki;
31     }
32     public void setMTytul_ksiazki(TTytul_ksiazki mTytul_ksiazki) {
33         this.mTytul_ksiazki = mTytul_ksiazki;
34     }
35     public Date getTermin() {
36         return null;
37     }
38     public void setTermin(Date termin) {
39     }
40 }
```

The screenshot shows an IDE window with the following components:

- Project Explorer:** Shows the project structure for 'Wypożyczalnia1app', including source packages (META-INF, wypożyczalnia1app) and test packages. Files include persistence.xml, TAplikacja.java, TEgzemplarz.java, TEgzemplarzController.java, TEgzemplarz_termin.java, TFabryka.java, TTytul_ksiazki.java, TTytul_ksiazkiController.java, and TTytul_ksiazki_na_kasecie.java.
- Navigator:** Shows the 'Members View' for the 'TEgzemplarz' class, listing methods such as getId(), setId(), getMTytul_ksiazki(), setMTytul_ksiazki(), getTermin(), and setTermin().
- Code Editor:** Displays the source code for TEgzemplarz.java. Red boxes highlight the @Entity annotation, the Serializable interface, the @Id, @GeneratedValue, and @ManyToOne annotations, and the getter and setter methods for mTytul_ksiazki.



Wstawienie do modułu typu Persistence Unit wszystkich klas typu Entity (15)



3. Tworzenie warstwy klienta, prezentacji

Integrowanie warstwy prezentacji i klienta
wykonane w technologii **Java Server Faces**

- z projektem warstwy biznesowej zawierającej model obiektowy
- oraz warstwą integracji

Tworzenie warstwy prezentacji pozwalającej na utworzenie pustych tabel w bazie danych skojarzonej projektem - persistence.xml (1)

- Należy założyć projekt typu **Web Application** - slajd(1), slajd(2), slajd(3)
- Należy projekt typu **Java Application** zawierający warstwę biznesową oraz integracji (w przykładzie Wypożyczalnia1app) połączyć z projektem **Web Application** – należy w zakładce **Projects** prawym klawiszem myszy kliknąć na nazwę projektu i wybrać opcję **Properties** - slajd (4). Zgodnie ze slajdem (5) należy wybrać opcję **Libraries** w części **Categories** formularza **Properties**, i następnie kliknąć na przycisk **Add Project** – w kolejnym formularzu wybrać projekt typu **Java Application** i zatwierdzić. Po powrocie do zakładki **Projects** w opcji **Libraries** pojawił się plik typu jar zawierający pakiety projektu **Wypożyczalnia1app** z klasami typu **Entity** oraz z warstwą integracji (slajd (6)).
Uwaga: Przy przenoszeniu projektu należy usunąć podobnie jak przy łączeniu projektów projekt typu JavaApplication z projektu typu WebApplication i ponownie połączyć projekty jak wyżej.
- Zgodnie ze slajdem (7) należy przeciągnąć w trybie **Design** komponent **Table** na stronę **Page1.jsp**, reprezentującą warstwę prezentacji. Należy kliknąć prawym klawiszem myszy na komponent i następnie kliknąć na pozycję **Add Binding Attribute (slajd 0)**
- Zgodnie ze slajdem (8) i slajdem(9), należy w klasie sesji typu **SessionBean1** wpisać kod pozwalający pobierać dane o tytułach z bazy danych i mapować je na obiekty typu **TTytul_książki** i **TTytul_książki_na_kasecie** zapisując je do tabeli **TTytul_książki_tytuly[]**. Wykonuje to metoda **updateTytuls()** wywołana w metodzie **init** klasy **SessionBean1**. Metoda **updateTytuls()** wywołuje metodę **getTTytul_książkis** z klasy **TTytul_książkiController**, czyli pobiera z bazy danych zawartość tabeli **TTytul_książki**. **Jeśli baza jest pusta, spowoduje wywołanie skryptów typu Create tworzących puste tabele w wyniku mapowania modelu obiektowego**
- Zgodnie ze slajdami (10) i (11), należy wykonać operację **Bind to Data**: kliknąć prawym klawiszem myszy komponent **Table**, wybrać opcję **Bind to Data**. Następnie w formularzu tej operacji w zakładce **Bind to object** wybrać z opcji **Get Data From** tablicę **tytuly** z klasy **SessionBean1**. Na slajdzie (13) pokazano rezultat działań.
- Należy uruchomić aplikację (Run) – slajd(14)

Tworzenie warstwy prezentacji pozwalającej na utworzenie pustych tabel w bazie danych skojarzonej projektem - persistence.xml (2)

- Zgodnie ze slajdem (15) w zakładce **Services** po wyborze bazy danych współpracującej z aplikacją (+) pojawiły się trzy tabele: **TTytul_książki** i **TEgzemplarz** – obie przechowują dane odpowiadające dwom typom danych dzięki dziedziczeniu w obu tych klasach, Kolumna **DTYPE** służy do rozróżniania typu tych danych w modelu relacyjnym. Trzecia tabela **SEQUENCE** przechowuje informacje do generowania nie powtarzających się wartości kluczy głównych dzięki adnotacji **@Id**
@GeneratedValue(strategy = GenerationType.AUTO) przy kluczach głównych w klasach typu **Entity**, określając strategię domyślną generowania kluczy głównych w technologii **TopLink**
Pokazano wynik działania opcji **Design Query** wybranej z wyskakującej listy po naciśnięciu prawego klawisza na nazwie tabeli **TEgzemplarz** i przeciągnięciu lewym klawiszem myszy tabeli **TTytul_książki**.
- Po uruchomieniu w oknie Output, w zakładce GlassFishV2 przedstawiono skrypty tworzenia pustych tabel w bazie danych (skrypty Create) przy drugim uruchomieniu Run (wygenerowane wyjątki są związane z próbą tworzenia ponownie tabel) – slajd (16). Na slajdzie (17) pokazano działanie aplikacji z wyłączoną opcją Create w pliku persistence.xml (włączono opcję None).
- **W dalszej części projektu należy sprawdzać komunikaty wyświetlane w zakładce GlassFishV2 okienka Output. Mogą pomóc w wykryciu błędów w aplikacji. Część błędów, mimo poprawy w kodzie aplikacji, można usunąć po zamknięciu serwera aplikacji – można to zrobić w zakładce Services w pozycji Servers lub przez ponowne uruchomienie środowiska NetBeans 6.01. Każda zmiana w dołączonym projekcie typu Java Application wymaga ponownego skompilowania tego projektu w trybie Clean and Build**

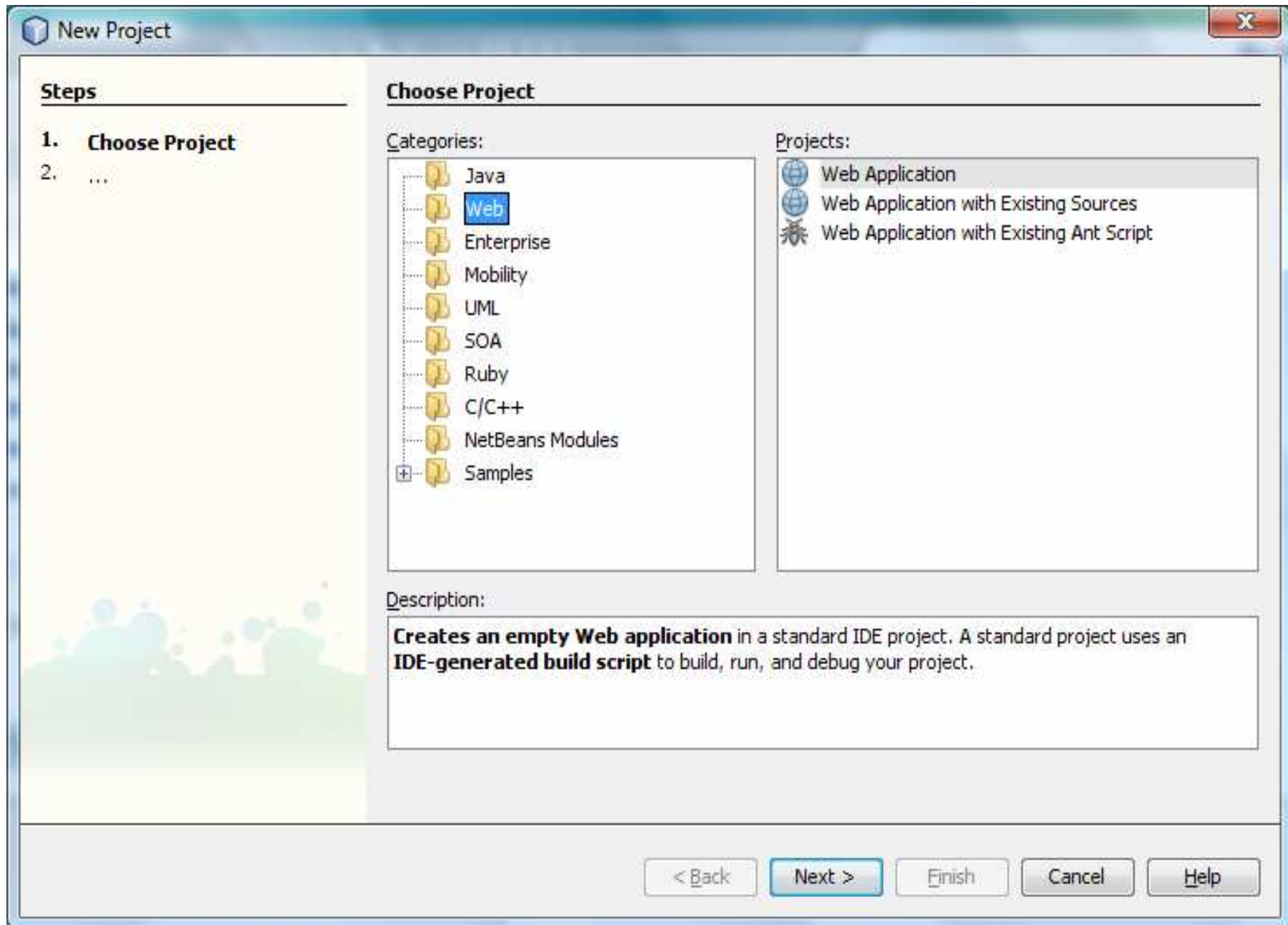
Wstawianie do pliku typu Java strony JSP obiektów reprezentujących komponenty wizualnego interfejsu graficznego użytkownika (0)

The screenshot shows an IDE window with a JSP page titled "Page1 *". The page contains a table with the following data:

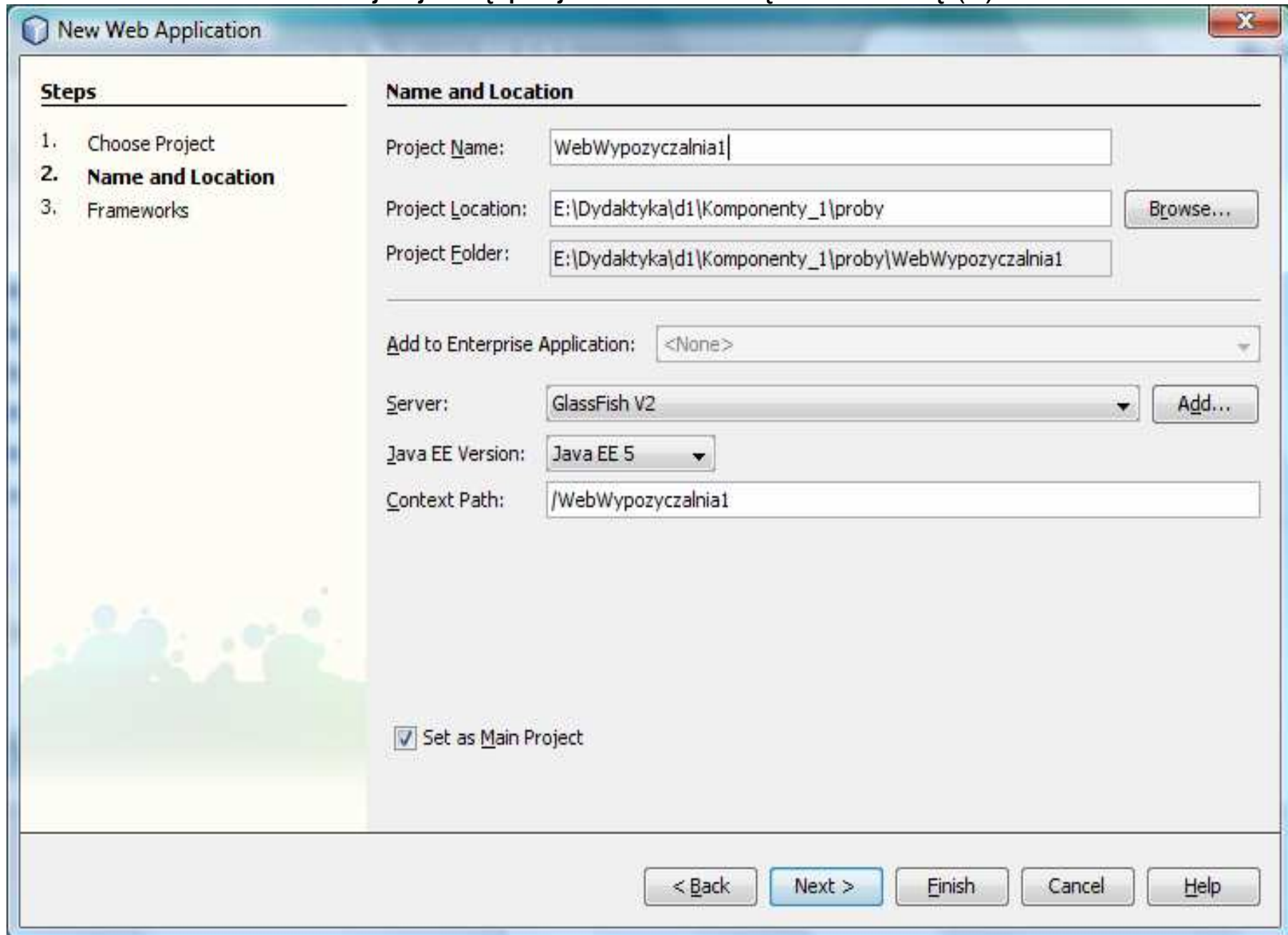
id	tytul	autor
123	abc	abc
123	abc	abc
123	abc	abc

A context menu is open over the table, with "Add Binding Attribute" selected. The menu items include: Edit Java Source, Table Layout..., Bind to Data..., Add Row Group, Property Bindings..., Add Binding Attribute, Edit JSP Source, Edit Inline, Select Parent, Edit Event Handler, Set Initial Focus, Virtual Forms..., Snap to Grid, Bring to Front, Send to Back, Customize, Cut (Ctrl+X), Copy (Ctrl+C), Paste (Ctrl+V), Delete, and Preview in Browser... The "Palette" on the right shows various UI components like Label, Text Field, Button, etc. The "Properties" window for "bazatytyly" shows the "id" property set to "bazatytyly".

Tworzenie projektu kategorii Web typu Web Application (1)



Projekt typu WebApplication tworzony w tym samym katalogu, w którym znajduje się projekt z warstwą biznesową (2)



The screenshot shows the 'New Web Application' wizard in an IDE. The window title is 'New Web Application'. The 'Steps' pane on the left shows three steps: 1. Choose Project, 2. Name and Location (which is the current step), and 3. Frameworks. The 'Name and Location' section contains the following fields and options:

- Project Name:** WebWypożyczalnia1
- Project Location:** E:\Dydaktyka\d1\Komponenty_1\proby (with a 'Browse...' button)
- Project Folder:** E:\Dydaktyka\d1\Komponenty_1\proby\WebWypożyczalnia1
- Add to Enterprise Application:** <None>
- Server:** GlassFish V2 (with an 'Add...' button)
- Java EE Version:** Java EE 5
- Context Path:** /WebWypożyczalnia1
- Set as Main Project

At the bottom of the wizard, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

Projekt powinien być oparty na Visual Web JavaServer Faces (3)

New Web Application

Steps

1. Choose Project
2. Name and Location
- 3. Frameworks**

Frameworks

Select the frameworks you want to use in your web application.

- Visual Web JavaServer Faces
- JavaServer Faces
- Struts 1.2.9

Visual Web JavaServer Faces Configuration

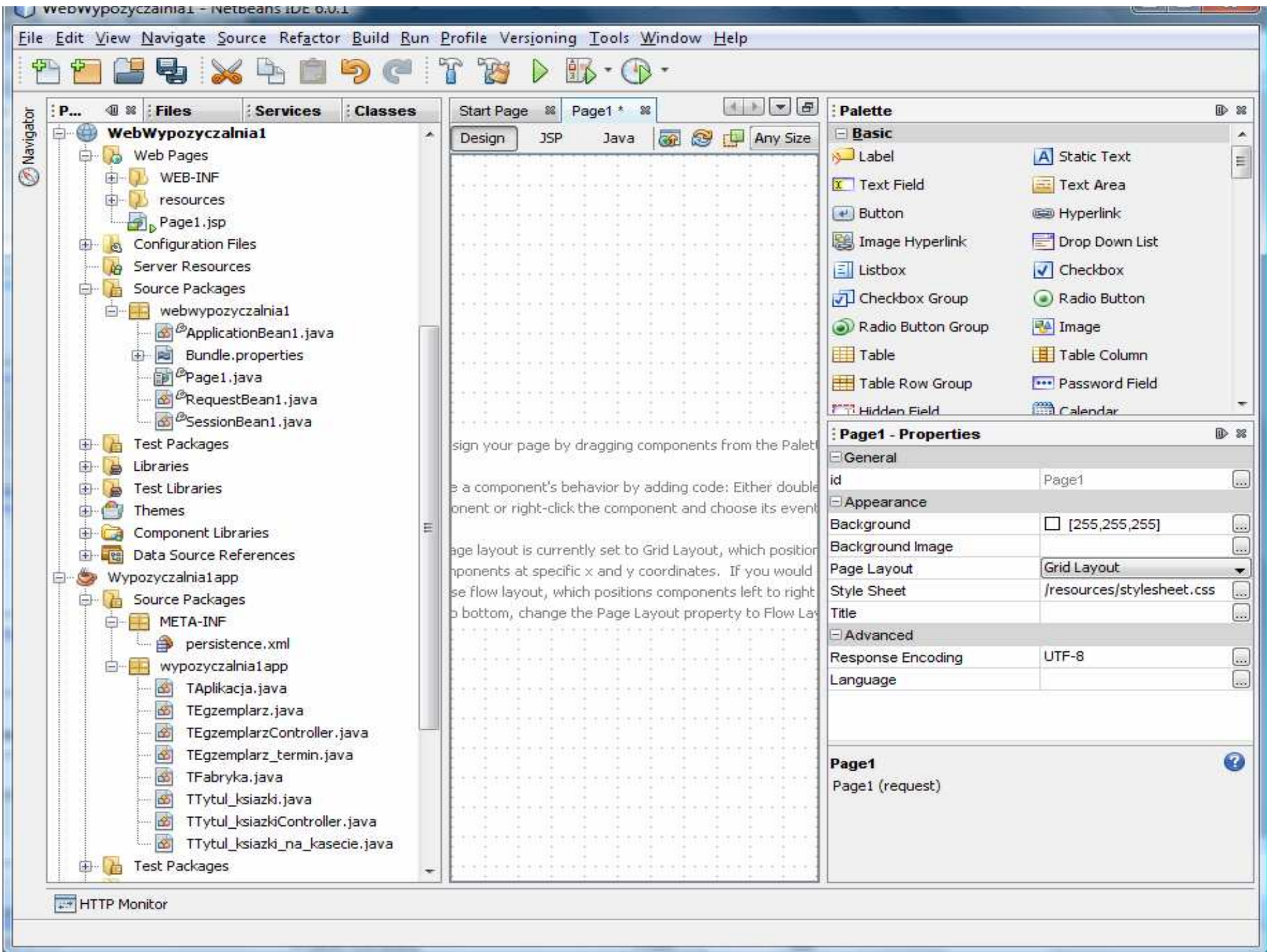
Default Java Package:

JSF Servlet Name:

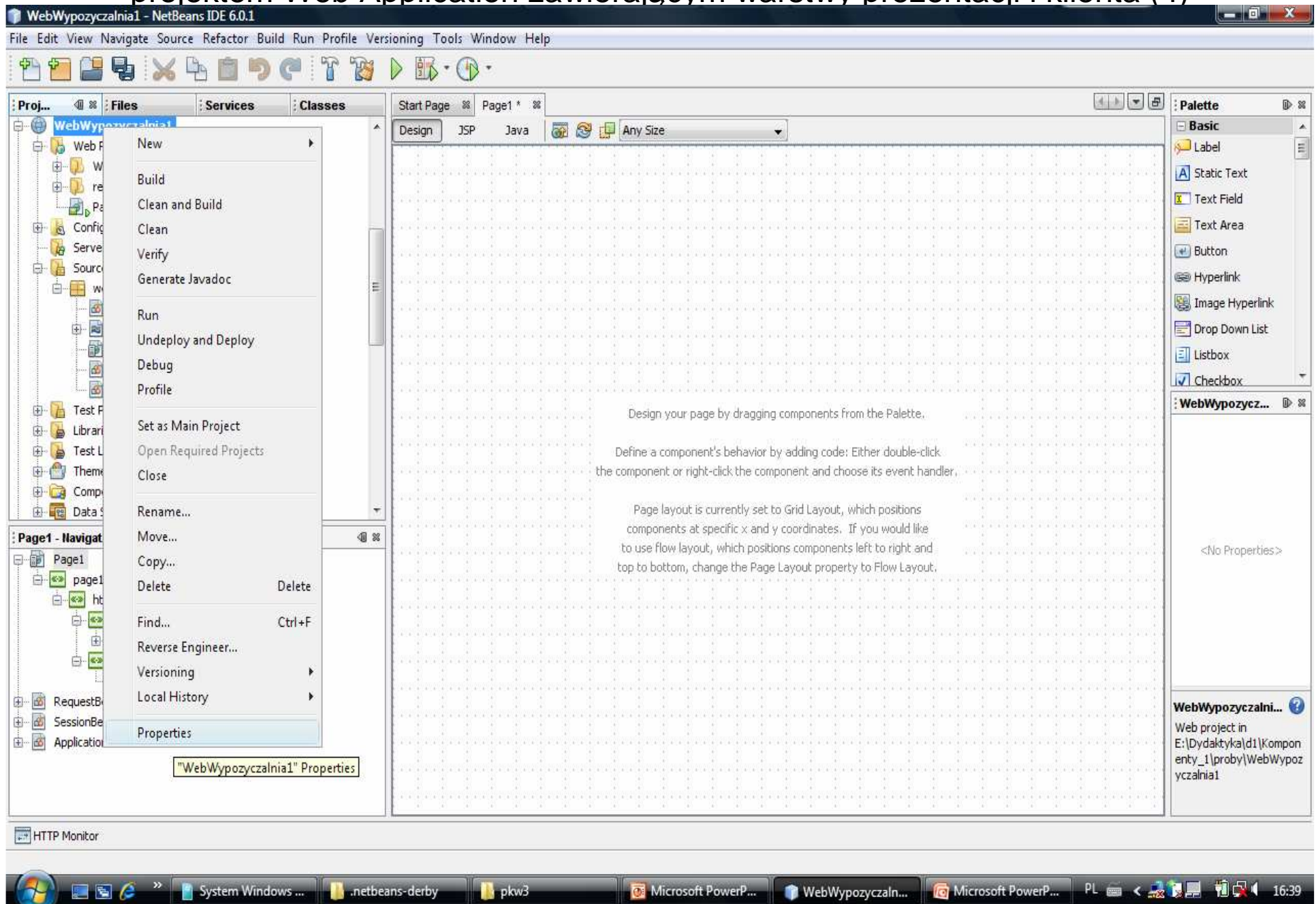
Servlet URL Mapping:

Validate XML Verify Objects

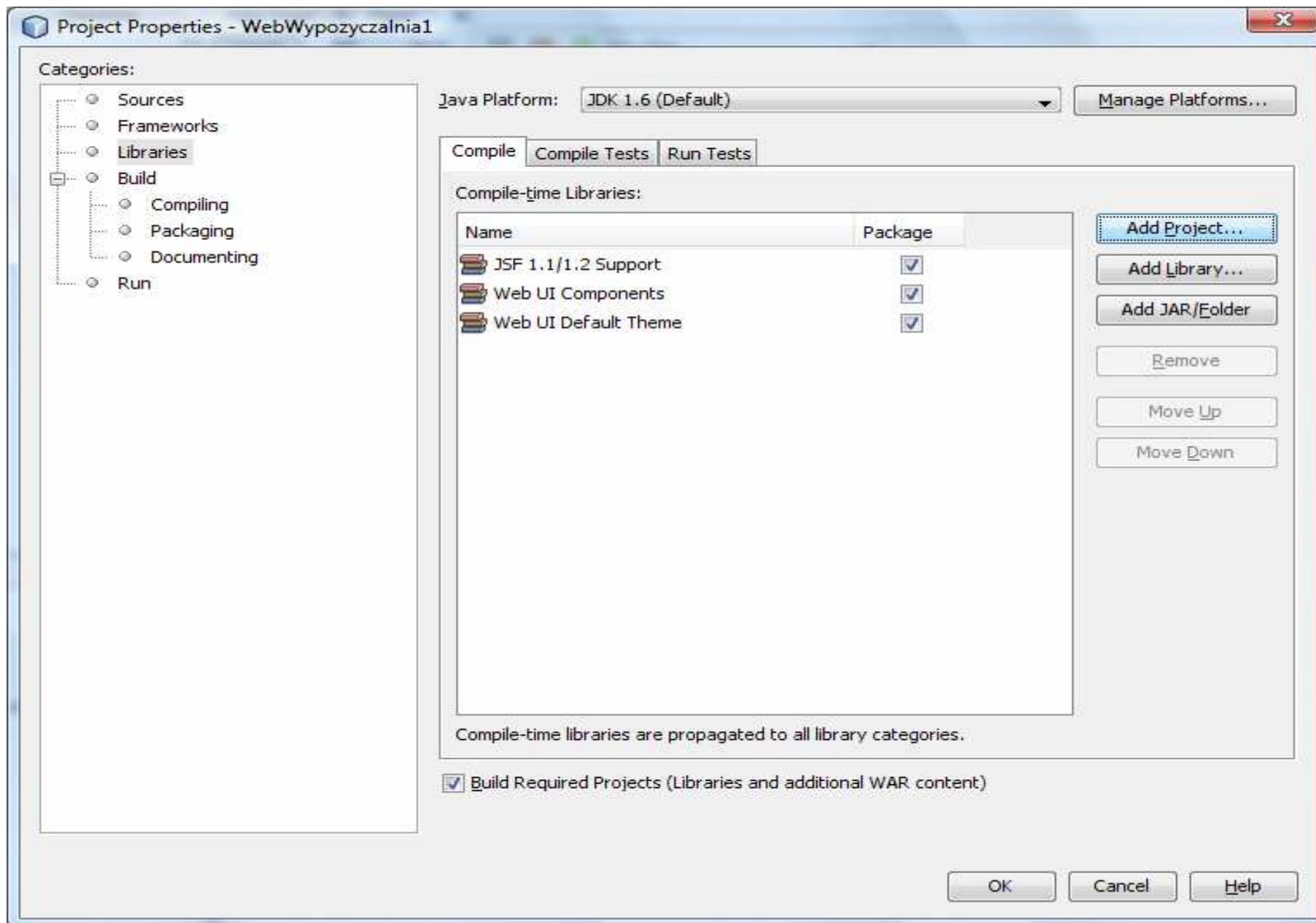
< Back Next > **Finish** Cancel Help

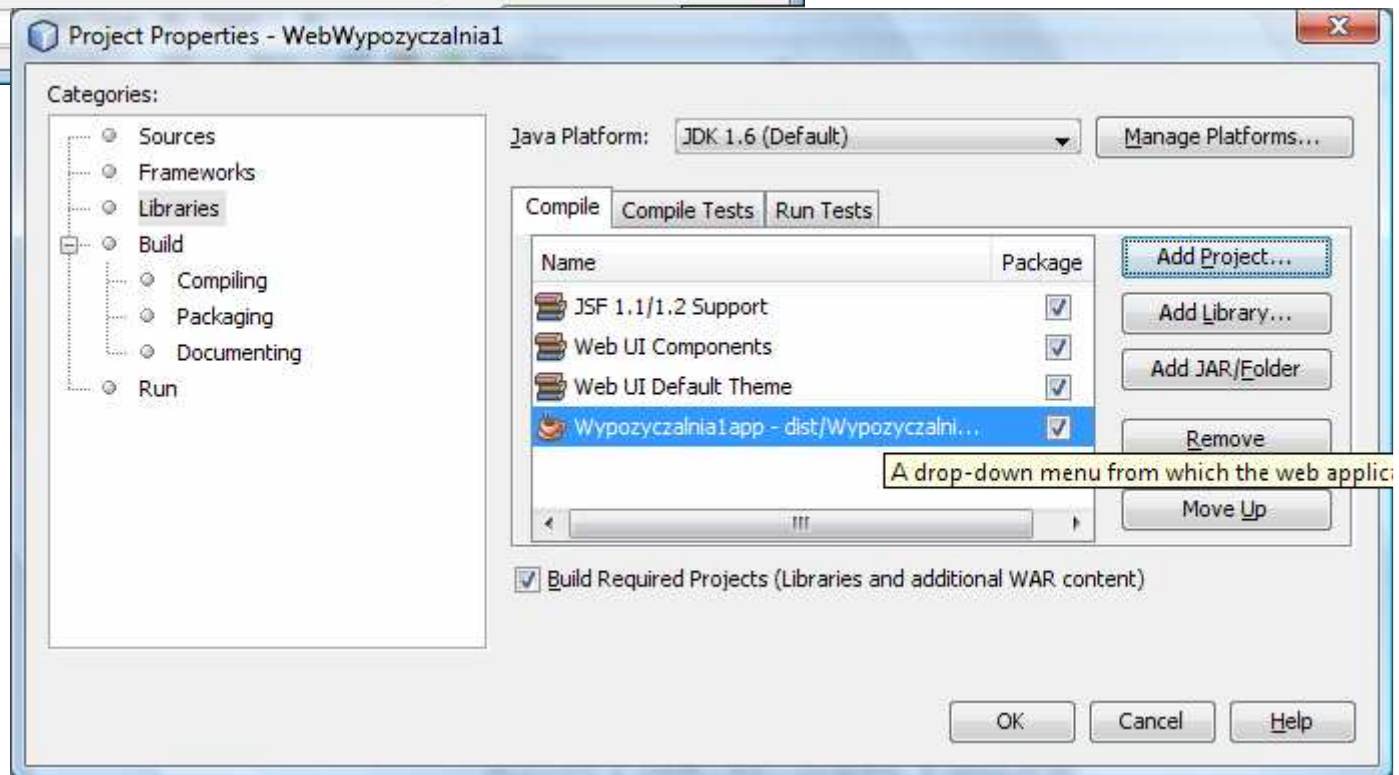
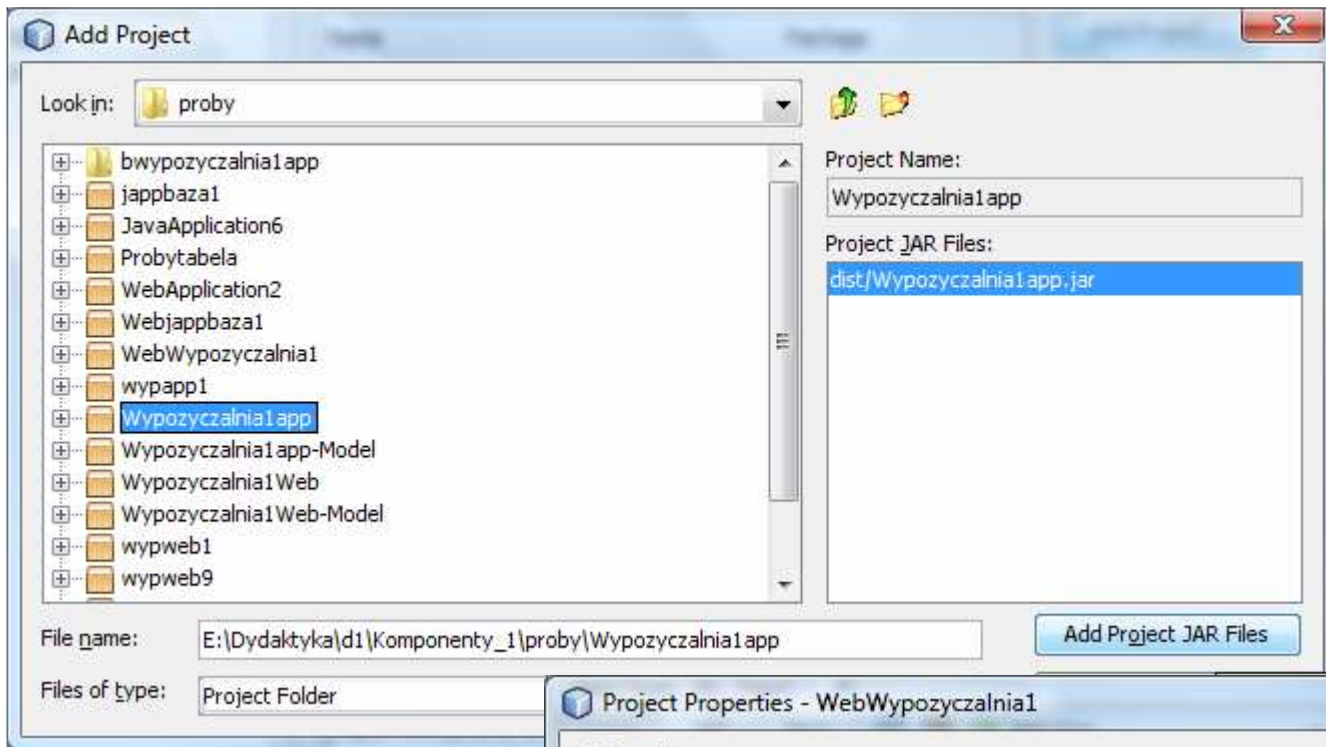


Łączenie projektu typu JavaApplication zawierającym warstwę biznesową i integracji z projektem Web Application zawierającym warstwy prezentacji i klienta (4)

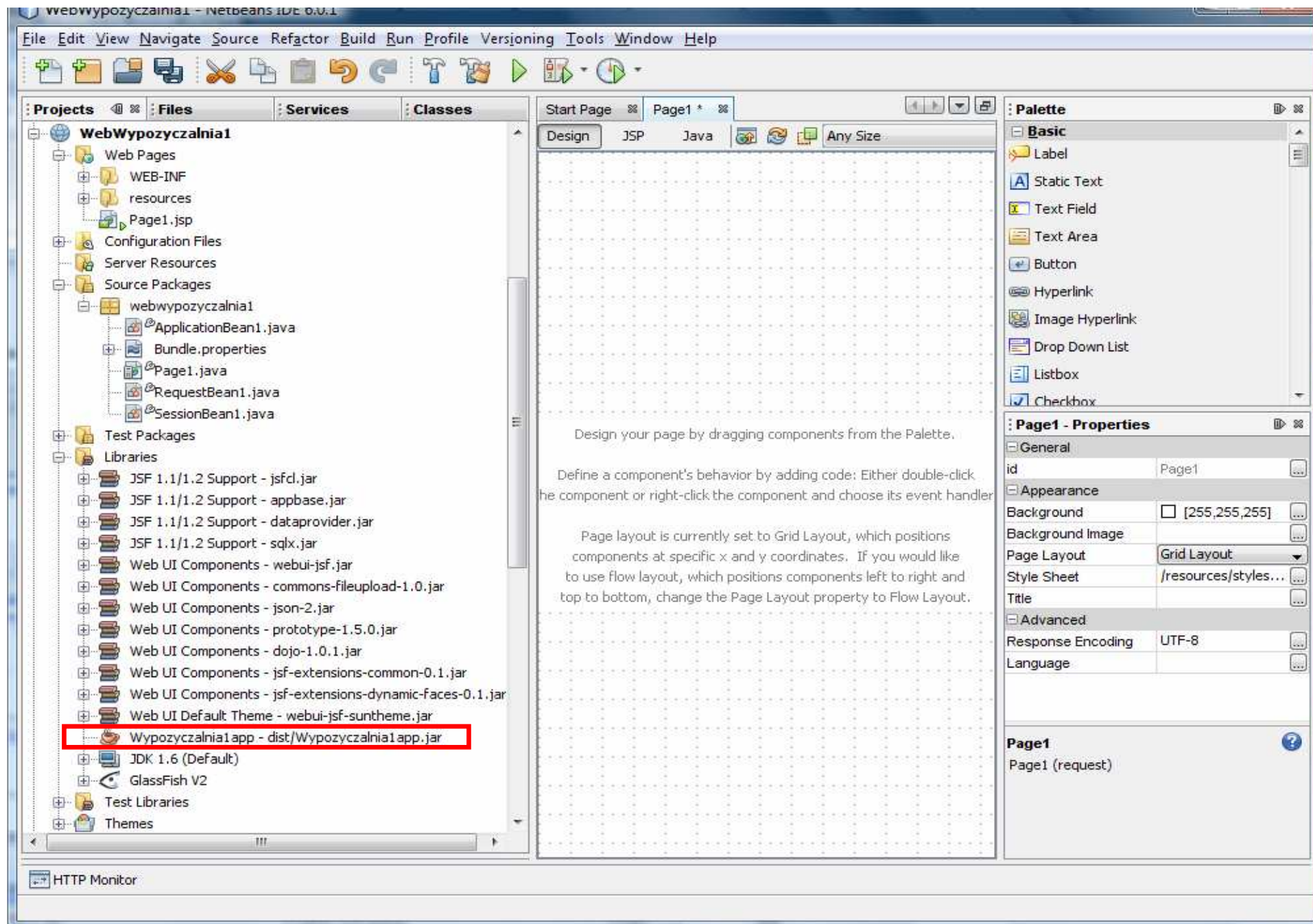


Wybór w okienku **Properties** projektu **Web Application** opcji **Libraries** oraz **Add Project (5)**





Połączone projekty – projekt Web Application korzysta z klas projektu Java Application (6)



Wstawienie do warstwy prezentacji (**Page1.jsp**) komponentu **Table** w celu wyświetlania w tabeli danych typu **TTytul_ksiazki** oraz **TTytul_ksiazki_na_kasecie** z bazy danych (7)

The screenshot shows the NetBeans IDE interface. The main window displays a JSP page in Design mode. A table component is placed on the page, containing the following data:

column1	column2	column3
row1_column1	row1_column2	row1_column3
row2_column1	row2_column2	row2_column3
row3_column1	row3_column2	row3_column3
row4_column1	row4_column2	row4_column3
row5_column1	row5_column2	row5_column3

The left sidebar shows the Project Explorer with the following structure:

- WebWypożyczalnia1
 - Web Pages
 - WEB-INF
 - resources
 - Page1.jsp
 - Configuration Files
 - Server Resources
 - Source Packages
 - webwypożyczalnia1
 - @ApplicationBean1.java
 - Bundle.properties
 - @Page1.java
 - @RequestBean1.java
 - @SessionBean1.java
 - Test Packages
 - Libraries
 - JSF 1.1/1.2 Support - jsfcl.jar
 - JSF 1.1/1.2 Support - appbase.jar
 - JSF 1.1/1.2 Support - dataprovider.jar

The bottom-left pane shows the 'table1 - Navigator' with the following structure:

- Page1
 - page1
 - html1
 - head1
 - body1
 - form1
 - table1
 - defaultTableDataProvider
 - RequestBean1
 - SessionBean1
 - ApplicationBean1

The right sidebar shows the 'Pal...' palette with various components, and the 'tab...' properties window for the selected table component, showing properties like 'id', 'appearance', 'augmentTi', 'cellPadding', 'cellSpacing', 'clearSortBu', 'deselectMut', 'deselectSin', 'footerText', 'itemsText', 'lite', 'paginateBut', and 'table1' (table1 (Table)).

Wpisanie kodu wywołującego kod warstwy integracji (8)

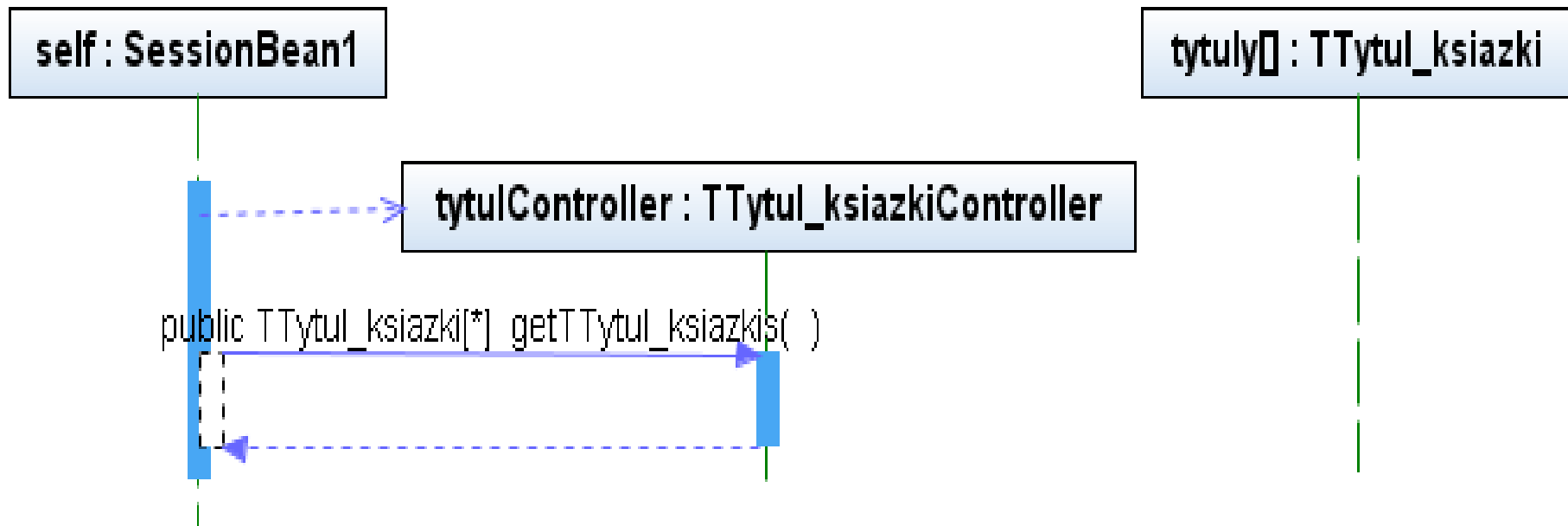
The screenshot shows the NetBeans IDE 6.0.1 interface. The left sidebar displays a project tree with 'WebWypożyczalnia1' selected. The main editor window shows the 'SessionBean1.java' file. The code is as follows:

```
55  */
56  @Override
57  public void init() {
58      // Perform initializations inherited from our superclass
59      super.init();
60      // Perform application initialization that must complete
61      // *before* managed components are initialized
62      // TODO - add your own initialization code here
63
64      Managed Component Initialization
65
66      // Perform application initialization that must complete
67      // *after* managed components are initialized
68      // TODO - add your own initialization code here
69
70  updateTytuls();
71  }
72  private TTytul_książki tytuly[];
73  public TTytul_książki[] getTytuly() {
74      return tytuly;
75  }
76
77  public void setTytuly(TTytul_książki[] tytuly) {
78      this.tytuly = tytuly;
79  }
80
81  public void updateTytuls() {
82      TTytul_książkiController tytulController = new TTytul_książkiController();
83      tytuly = tytulController.getTTytul_książkis();
84  }
```

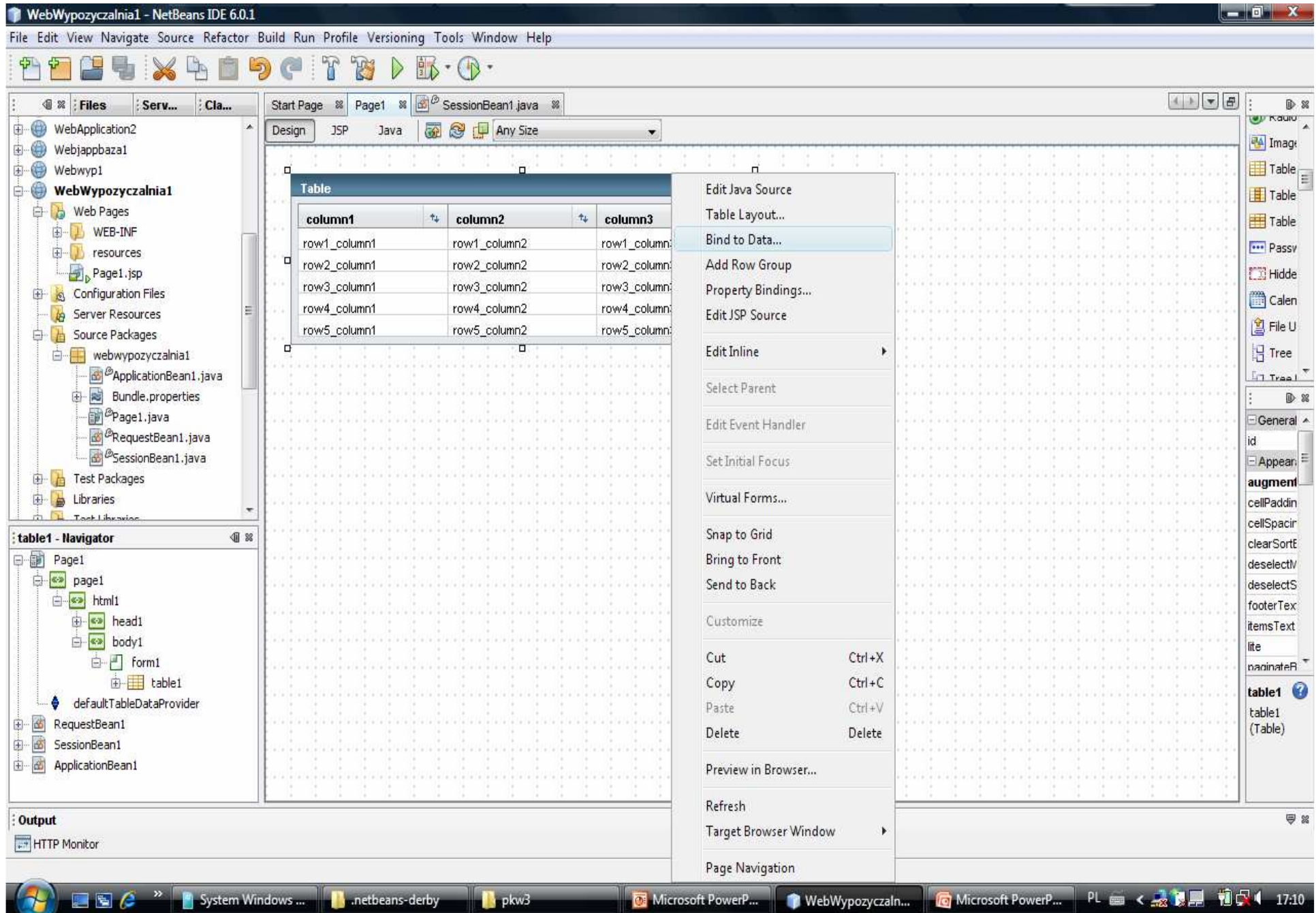
The code from line 70 to 84 is highlighted with a red box. The status bar at the bottom shows '78:20 INS'. The bottom panel shows the 'Output' window with tabs for 'Java DB Database Process', 'GlassFish V2', and 'WebWypożyczalnia1 (run)'. The 'HTTP Monitor' window is also visible at the bottom.

Odczyt tytułów z bazy danych (9)

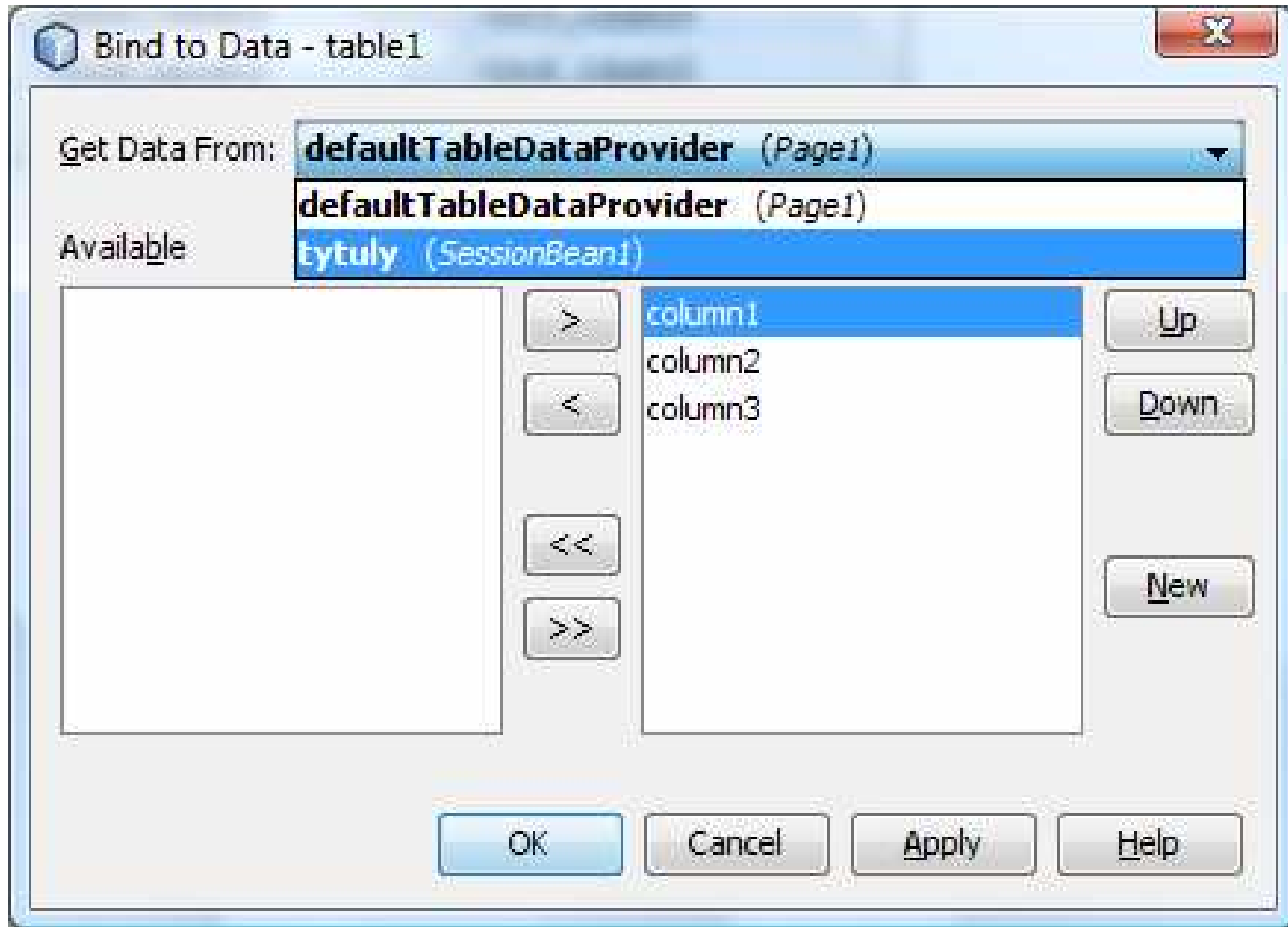
public void SessionBean1::updateTytuls()



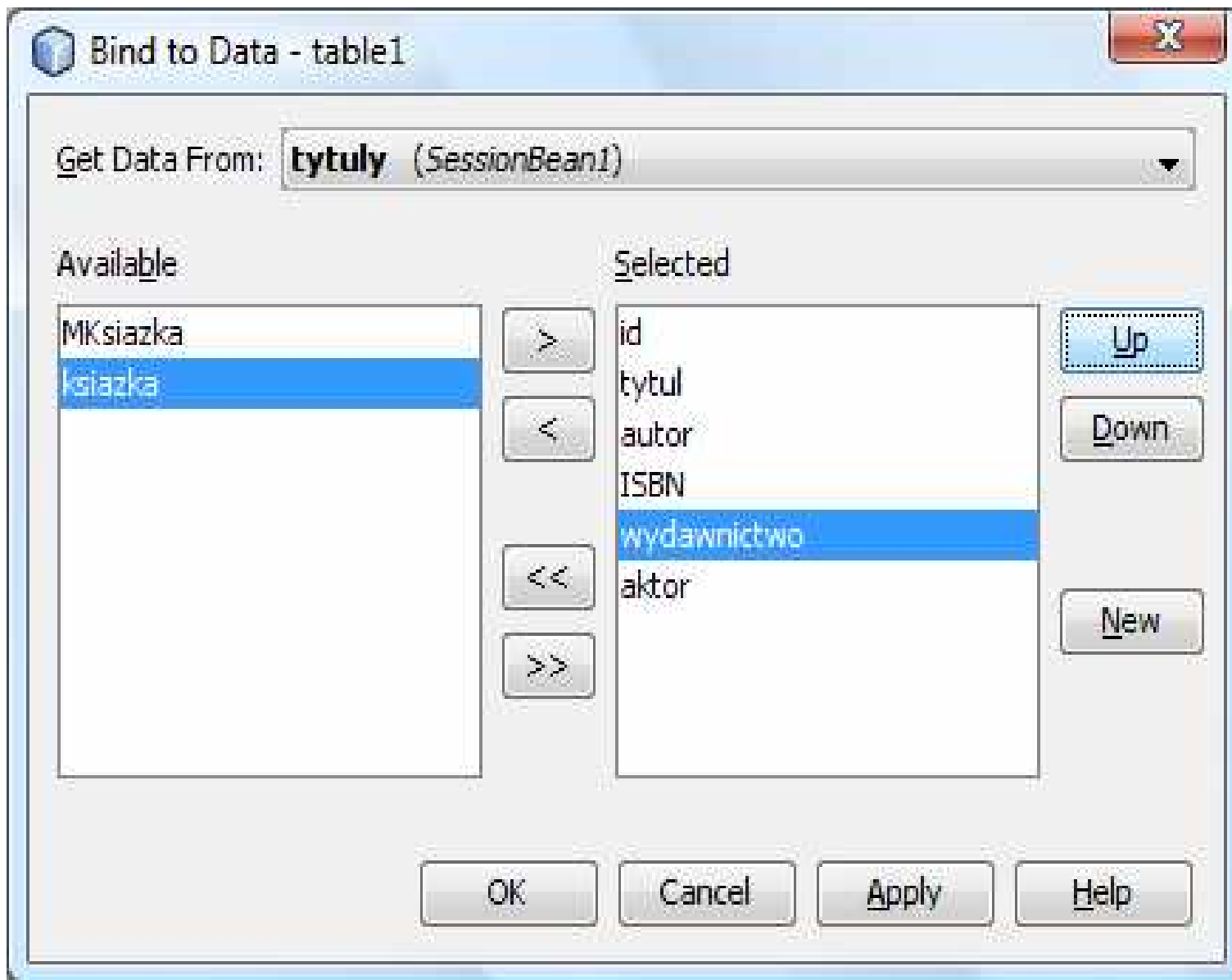
Łączenie komponentu **Table** z tablicą **Ttytul_książki książki[]** – wybór **Bind to Data (10)**



Wybór tablicy **tytuly** w okienku **Bind to Data (11)**



Wybór kolumn (pomijanie kolekcji książka mapującej relację **OneToMany** oraz **MKsiążka** wykorzystywanej przez warstwę biznesową do gromadzenia danych o książkach dla danego tytułu) (12)



Efekt operacji Bind to Data (13)

The screenshot displays the NetBeans IDE 6.0.1 interface. The main design view shows a table component with the following data:

id	tytuł	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

The interface also shows a File Explorer on the left, a Palette on the right, and an Output window at the bottom.

Uruchomienie aplikacji (14)

The screenshot shows a Windows Internet Explorer browser window. The address bar displays the URL `http://localhost:8080/WebWypożyczalnia1/`. The menu bar includes "Plik", "Edycja", "Widok", "Ulubione", "Narzędzia", and "Pomoc". The toolbar contains various icons for home, search, and navigation. A Norton security notification bar is visible, stating "Monitorowanie fałszywych witryn jest włączone". The main content area displays a table with the following structure:

Table					
id	tytuł	autor	ISBN	wydawnictwo	aktor
No items found.					

The status bar at the bottom shows "Goto", "Internet | Tryb chroniony: włączony", and a zoom level of "100%".

Wygenerowano tabele w bazie danych **katalog**, założonej wcześniej do utrwalania

The screenshot displays the NetBeans IDE 6.0.1 interface. On the left, the 'Projects' pane shows a database connection 'jdbc:derby://localhost:1527/katalog [kruk on KRUK]' with a tree view of tables and foreign keys. The 'TEGZEMPLARZ' table is highlighted in red. The 'TTYTUL_KSIAZKI' table is also visible. The main workspace shows a diagram of the two tables with a foreign key relationship. Below the diagram is a table listing columns and their aliases. At the bottom, a SQL query editor contains a query that selects all columns from both tables, joined by their primary keys.

Column	Alias	Table	Output	Sort Type
ID		KRUK.TEGZEMPLARZ	<input checked="" type="checkbox"/>	
DTYPE		KRUK.TEGZEMPLARZ	<input checked="" type="checkbox"/>	
NUMER		KRUK.TEGZEMPLARZ	<input checked="" type="checkbox"/>	
MTYTUL_KSIAZKI_ID		KRUK.TEGZEMPLARZ	<input checked="" type="checkbox"/>	

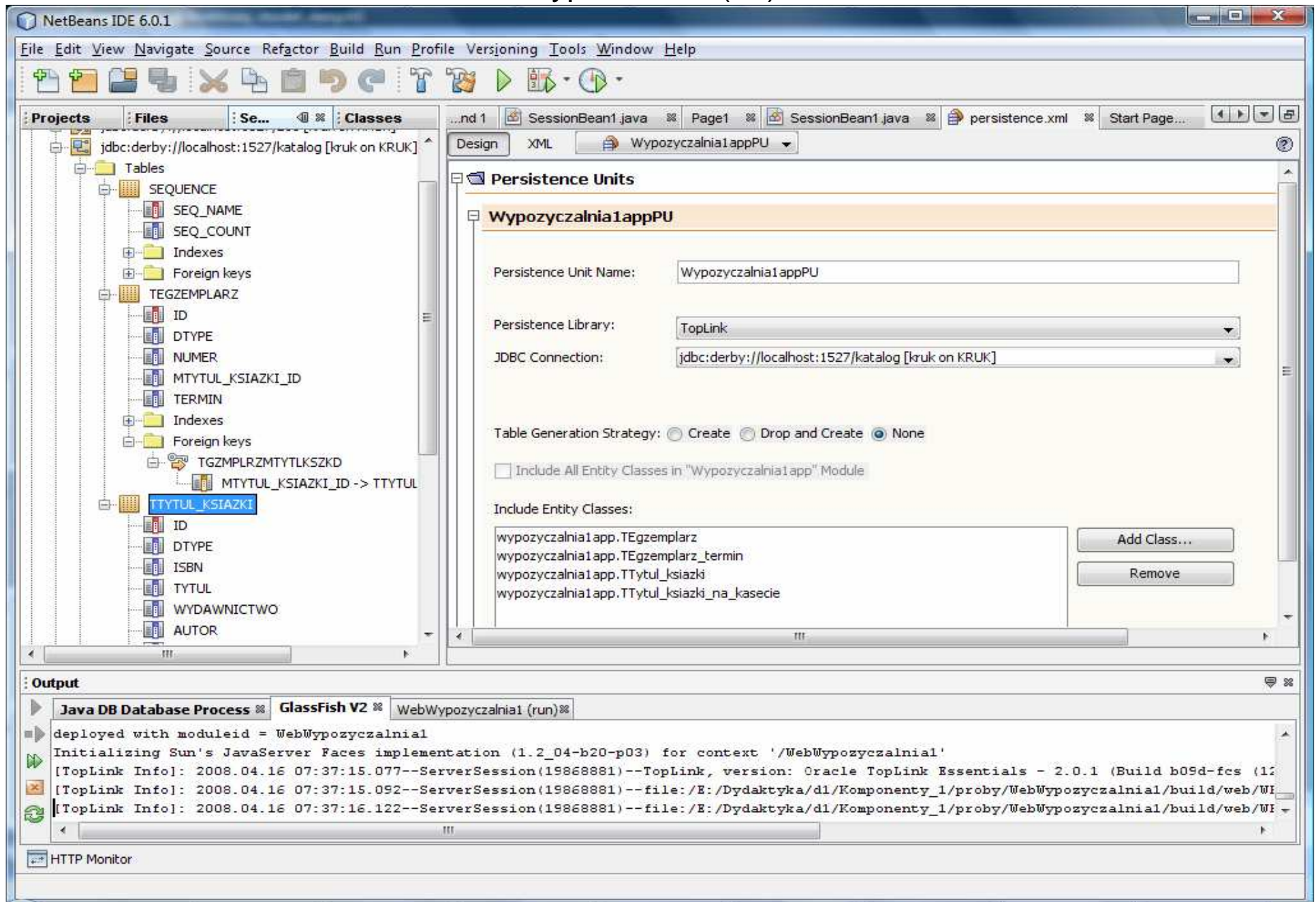
```
SELECT ALL KRUK.TEGZEMPLARZ.ID,  
          KRUK.TEGZEMPLARZ.DTYPE,  
          KRUK.TEGZEMPLARZ.NUMER,  
          KRUK.TEGZEMPLARZ.MTYTUL_KSIAZKI_ID,  
          KRUK.TEGZEMPLARZ.TERMIN,  
          KRUK.TTYTUL_KSIAZKI.ID,  
          KRUK.TTYTUL_KSIAZKI.DTYPE,  
          KRUK.TTYTUL_KSIAZKI.ISBN,  
          KRUK.TTYTUL_KSIAZKI.TYTUL,  
          KRUK.TTYTUL_KSIAZKI.WYDAWNICTWO,  
          KRUK.TTYTUL_KSIAZKI.AUTOR,  
          KRUK.TTYTUL_KSIAZKI.AKTOR  
FROM KRUK.TEGZEMPLARZ  
INNER JOIN KRUK.TTYTUL_KSIAZKI  
ON KRUK.TEGZEMPLARZ.MTYTUL_KSIAZKI_ID = KRUK.TTYTUL_KSIAZKI.ID
```

Ponowne wywołanie aplikacji w trybie **Create** (16)

The screenshot shows the NetBeans IDE interface with the following components:

- Projects:** WebWypożyczalnia
- Files:** Web Pages, Configuration Files, Server Resources
- Classes:** SessionBean1.java, persistence.xml, Start Page, jdbc:derby://localhost:1527/katalog [kruk on KRUK]
- Code Editor:** Contains a SQL query: `SELECT ALL KRUK.TEGZEMPLARZ.ID,`
- Output:** Shows the deployment process and subsequent errors. The errors are highlighted with a red box and include:
 - Initializing Sun's JavaServer Faces implementation (1.2_04-b20-p03) for context '/WebWypożyczalnia'
 - [TopLink Info]: 2008.04.16 05:39:41.226--ServerSession(10882451)--TopLink, version: Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))
 - [TopLink Info]: 2008.04.16 05:39:41.679--ServerSession(10882451)--file:/E:/Dydaktyka/dl/Komponenty_1/proby/WebWypożyczalnia/build/web/WEB-INF/lib/Wypożyczalniaapp.jar-Wypoży
 - [TopLink Info]: 2008.04.16 06:39:41.961--ServerSession(10882451)--file:/E:/Dydaktyka/dl/Komponenty_1/proby/WebWypożyczalnia/build/web/WEB-INF/lib/Wypożyczalniaapp.jar-Wypoży
 - Initializing Sun's JavaServer Faces implementation (1.2_04-b20-p03) for context '/WebWypożyczalnia'
 - [TopLink Info]: 2008.04.16 07:24:08.837--ServerSession(28940314)--TopLink, version: Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))
 - [TopLink Info]: 2008.04.16 07:24:08.852--ServerSession(28940314)--file:/E:/Dydaktyka/dl/Komponenty_1/proby/WebWypożyczalnia/build/web/WEB-INF/lib/Wypożyczalniaapp.jar-Wypoży
 - [TopLink Warning]: 2008.04.16 07:24:08.899--ServerSession(28940314)--Exception [TOPLINK-4002] (Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))): oracle.toplink
 - Internal Exception: java.sql.SQLException: Table/View 'TEGZEMPLARZ' już istnieje w Schema 'KRUK'.
 - Error Code: -1
 - Call: CREATE TABLE TEGZEMPLARZ (ID BIGINT NOT NULL, DTYPE VARCHAR(31), NUMER INTEGER, MTYTUL_KSIAZKI_ID BIGINT, TERMIN DATE, PRIMARY KEY (ID))
 - Query: DataModifyQuery()
 - [TopLink Warning]: 2008.04.16 07:24:08.915--ServerSession(28940314)--Exception [TOPLINK-4002] (Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))): oracle.toplink
 - Internal Exception: java.sql.SQLException: Table/View 'TTYTUL_KSIAZKI' już istnieje w Schema 'KRUK'.
 - Error Code: -1
 - Call: CREATE TABLE TTYTUL_KSIAZKI (ID BIGINT NOT NULL, DTYPE VARCHAR(31), ISBN VARCHAR(255), TYTUL VARCHAR(255), WYDAWNICTWO VARCHAR(255), AUTOR VARCHAR(255), AKTOR VARCHAR(25
 - Query: DataModifyQuery()
 - [TopLink Warning]: 2008.04.16 07:24:08.946--ServerSession(28940314)--Exception [TOPLINK-4002] (Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))): oracle.toplink
 - Internal Exception: java.sql.SQLException: Constraint 'TGZEMPLRZMTYTLKSZKD' już istnieje w Schema 'KRUK'.
 - Error Code: -1
 - Call: ALTER TABLE TEGZEMPLARZ ADD CONSTRAINT TGZEMPLRZMTYTLKSZKD FOREIGN KEY (MTYTUL_KSIAZKI_ID) REFERENCES TTYTUL_KSIAZKI (ID)
 - Query: DataModifyQuery()
 - [TopLink Warning]: 2008.04.16 07:24:08.962--ServerSession(28940314)--Exception [TOPLINK-4002] (Oracle TopLink Essentials - 2.0.1 (Build b09d-fcs (12/06/2007))): oracle.toplink
 - Internal Exception: java.sql.SQLException: Table/View 'SEQUENCE' już istnieje w Schema 'KRUK'.
 - Error Code: -1
 - Call: CREATE TABLE SEQUENCE (SEQ_NAME VARCHAR(50) NOT NULL, SEQ_COUNT DECIMAL, PRIMARY KEY (SEQ_NAME))
 - Query: DataModifyQuery()

Ponowne uruchamianie aplikacji po założeniu tabel bez wykonywania skryptów SQL typu Create (17)



4. Tworzenie warstwy klienta, prezentacji cd

Integrowanie warstwy prezentacji i klienta
wykonane w technologii **Java Server Faces**

- Wykonanie przypadku użycia „dodaj tytuł”

Tworzenie warstwy prezentacji dla przypadku użycia „dodaj tytuł”(1)

- Slajd (1) - z palety **Layout** należy przeciągnąć komponent **Grid Panel** i nadać mu **id = panel** w okienku **Properties**. Przeciągnąć komponent **Table** do tego panela oraz kolejny **Grid Panel** i nadać mu **id = bazaoperacjePanel** oraz właściwość **columns** równą 2 w okienku **Properties**. Do zagnieżdżonego panela wstawić dwa przyciski typu **Button**. Nazwy widoczne w trybie **Design** na komponentach wpisać do właściwości **text** lub **title** w okienkach **Properties** tych komponentów – **komponenty można wstawiać z palety „przeciągając” je i „upuszczając” w oknie Navigator**.
- **Należy kliknąć prawym klawiszem myszy na wstawiane komponenty w Okienku Navigator lub Editor i następnie kliknąć na pozycję Add Binding Attribute (punkt 3, slajd 0)**
- Slajd(2) - wstawić kolejny **Grid Panel** zagnieżdżony w panelu **panel** i nadać mu **id = tytułyaplikacjaPanel** oraz **columns=2** w okienku **Properties**. Wstawić pięć par komponentów **Label** i **Text Field**. Nazwy widoczne w trybie **Design** na komponentach wpisać do właściwości **text** lub **Title** w okienkach **Properties** tych komponentów.
- Slajd (4) – zmodyfikować zawartość panela **panel**. Wstawić **Grid Panel** do panela **panel**, nadać mu **id=bazaPanel**, przesunąć komponent **Table** do tego panela i nadać mu **id=bazatytuły**, **columns=2** i **title=Tytuły** (okno **Properties**). Wstawić kolejny **Grid Panel** do panela **panel** i nadać mu **id=aplikacjadanePanel**. Przesunąć panel o **id= tytułyaplikacjaPanel** zagnieżdżony w panelu **panel** do panela **aplikacjadanePanel**. Wstawić kolejny **Grid Panel** do panela **aplikacjadanePanel**, nadać mu **id=aplikacjadane**, ustawić właściwość **columns** równą 2, przeciągnąć do niego komponent typu **DropDown List** i nadać mu **id = tytuły**. Wstawić kolejny **Grid Panel** zagnieżdżony w panelu **panel**, nadać mu **id = aplikacjaoperacjePanel**, ustawić właściwość **columns** równą 2 i przeciągnąć do niego komponent **Button**. Rozmieszczenie paneli ilustruje slajd (4). Uzupełnij właściwości **id**, **text** lub **Title** komponentów **Button** i **Text Field** w ich okienkach **Properties** zgodnie z slajdem (4).
- Napisać kod wg slajdu (3) w klasie **SessionBean1**. Wg slajdu (5) wykonać operację **Bind to Data**, łącząc komponent **tytuły** typu **DropDown List** z tablicą **Option tytuły_[]**.

Tworzenie warstwy prezentacji dla przypadku użycia „dodaj tytuł”(2)

- Napisać kod w klasie w **SessionBean1** (slajdy (6) i (7)), oznaczony ramką – oznacza ono **połączenie obiektu zdalnego** w postaci obiektu klasy **SessionBean1** do warstwy **biznesowej (obiekt aplikacja)**. Slajd (7) pokazuje metodę **przygotuj_tytuly()**, która pobiera dane o tytułach z obiektu **aplikacja (punkt 1)** i zapisuje je do tablicy **Option tytulys[] (punkty 2 i 3)**.
- W klasie **Page1** napisać kod wg slajdów (8) i (9). Na slajdzie (8) podano kod obsługi zdarzenia klikania na przycisk **dodajtytul_action()**. Kod ten służy do pobrania danych z komponentów panela **tytulyaplikacjaPanel (punkty 1, 2)** – w punkcie 1 ustala się zawartość pierwszego elementu tablicy dane, określające, jaki typ obiektu ma powstać 0 - obiekt typu **TTytul_książki**, 3 - obiekt typu **TTytul_książki_na_kasce**, w **punkcie 2** ustala się zawartość tablicy **dane** (obiekt typu **Transfer Object**), zapisu danych o tytułach w aplikacji metodą **dodaj_tytul** (w warstwie biznesowej) (**punkt 3**) i wyświetlenia danych o tytułach w komponentcie **tytulys** typu **DropDownList** za pośrednictwem tablicy **Option tytulys []** metodą **przygotujtytulys()** – (**punkt 4**). Slajd (9) pokazuje zawartość metody **prerender**, która zawsze jest wywoływana przy odświeżaniu zawartości strony w fazie **Response** przetwarzania strony **Java ServerFaces**. W tej metodzie wpisuje się obiekty typu String z pustym łańcuchem do komponentów typu Text Field oraz odświeża się zawartość komponentu Table (bazatytulys)
- Slajd (10) pokazuje stan aplikacji po uruchomieniu programu
- Zgodnie ze slajdem (11) napisać w klasie **SessionBean1** kod zapisujący dane aplikacji do bazy danych metodą **addTTytul_książkis** obiektu klasy **TTytul_książkiController** – **punkt 2**. Na slajdzie (13) pokazano diagram sekwencji tej operacji. Zapis do bazy następuje po wywołaniu metody **findTTytul_książkis** z klasy **TTytul_książkiController** (slajd (12)), która zapobiega naruszeniu więzów integralności tabeli **TTytul_książki** – **punkt 1**. Na slajdzie (14) pokazano działającą aplikację, która zapisuje dane o tytułach w aplikacji, wyświetla dane przechowywane w aplikacji za pomocą komponentu **tytulys** typu **Drop Down List**, zapisuje te dane do bazy danych i wyświetla zawartość tabeli **TTytul_książki** za pomocą tablicy **TTytul_książki tytulys []** w komponentcie typu **Table**
- Na slajdzie (15) pokazano zawartość danych bezpośrednio w tabeli **TTytul_książki** w bazie danych (zakładka **Service**) – po kliknięciu prawym klawiszem myszy na nazwę tabeli **TTytul_książki** wywołano z wyskakującego menu pozycję **View Data**.

Wstawianie komponentów (1)

The screenshot displays the NetBeans IDE 6.0.1 interface for a web application named "WebWypożyczalnia1". The main workspace is in Design mode, showing a JSP page with a table and two buttons. The table has the following data:

id	tytuł	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

Below the table, there are two buttons: "Zapisz tytuły do bazy" and "Zapisz egzemplarze do bazy".

The interface includes several panels:

- Project Explorer:** Shows the project structure, including "WebWypożyczalnia1" and its sub-components like "Web Pages", "Configuration Files", and "Server Resources".
- Palette:** Lists various UI components such as Label, Text Field, Button, Image Hyperlink, Listbox, Checkbox, Radio Button, and Image.
- Properties:** Shows the properties for the selected component, "bazaoperacjePanel", including General and Appearance settings.
- Output:** Displays the output of the application, including "Java DB Database Process", "GlassFish V2", and "WebWypożyczalnia1 (run)".

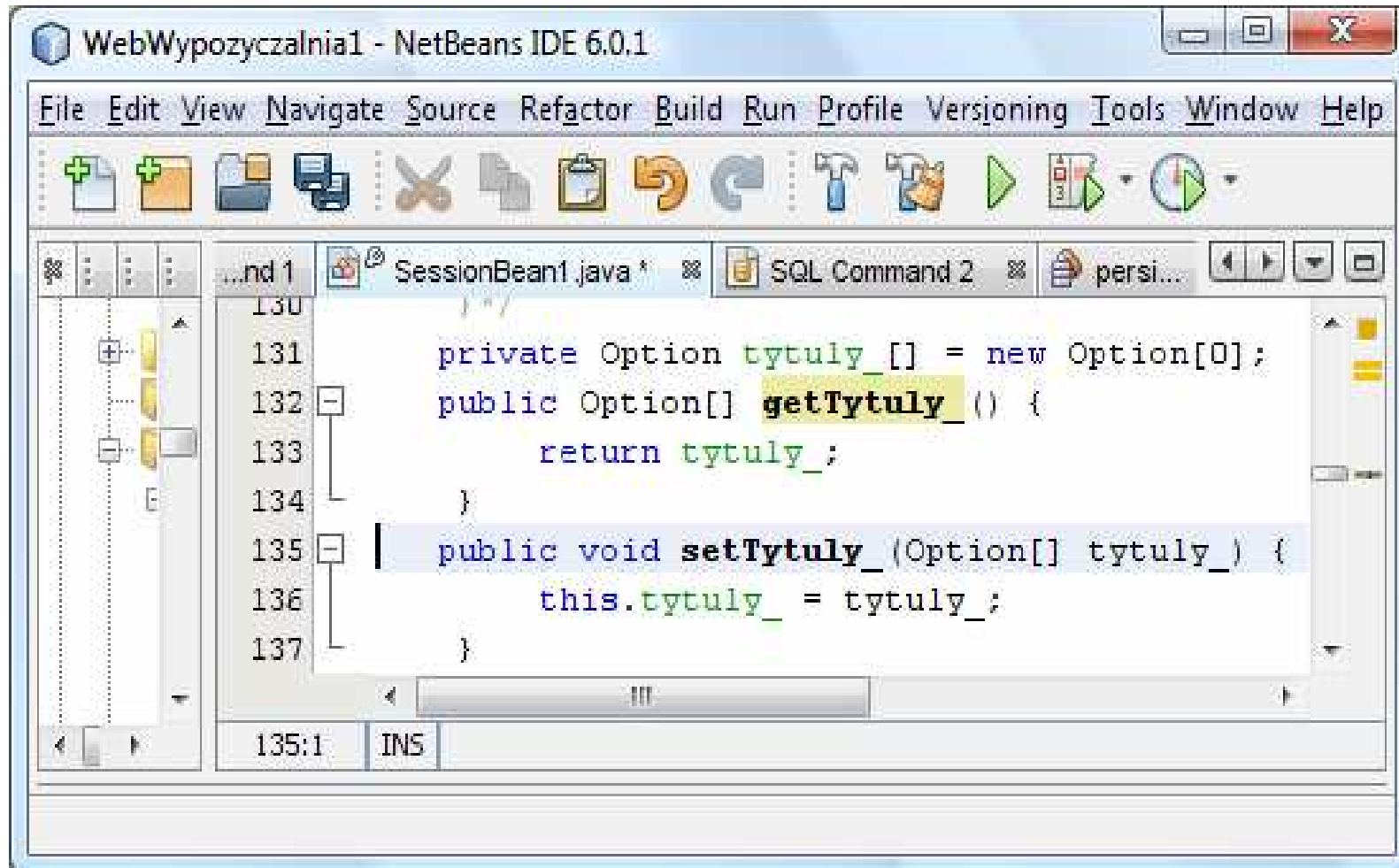
Wstawianie komponentów umieszczanych na komponencie **Grid Panel** z palety **Layout** (2)

The screenshot shows the NetBeans IDE 6.0.1 interface. The main editor is in Design mode, displaying a web page layout. At the top, there is a menu bar (File, Edit, View, Navigate, Source, Refactor, Build, Run, Profile, Versioning, Tools, Window, Help) and a toolbar. Below the toolbar, the 'Files' and 'Classes' panes show the project structure. The 'tytulyaplikacjaPanel - Navigator' pane shows a tree view with 'body1' containing 'form1', which contains 'panel'. The 'panel' contains a 'table1' with the following data:

id	tytul	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

Below the table are two buttons: 'Zapisz tytuły do bazy' and 'Zapisz egzemplarze do bazy'. Under each button are several input fields with labels: 'Podaj tytuł', 'Podaj autora', 'Podaj ISBN', 'Podaj wydawnictwo', and 'Podaj aktora'. The right sidebar contains the 'Palette' window with a 'Basic' category showing components like Label, Static Text, Text Area, Text Field, Button, Hyperlink, and Image Hyperlink. Below the palette is the 'tytulyaplikacjaPanel - Properties' window, showing a table of properties for the 'tytulyaplikacjaPanel' component, including 'id', 'tytulyaplikacj...', 'Appearance', 'bgcolor', 'border', 'cellpadding', 'cellspacing', 'columnClasses', 'columns' (set to 2), 'footerClass', and 'headerClass'. The bottom status bar shows the application is running, with 'Java DB Database Process', 'GlassFish V2', and 'WebWypożyczalnia1 (run)' visible.

Dodanie tablicy **Option tytuly_[]** w klasie sesji **SessionBean1** do wykonania operacji **Bind to Data** dla komponentu **tytuly** typu **DropDownList** w celu wyświetlenia tytułów przechowywanych w aplikacji (3)



The screenshot shows the NetBeans IDE 6.0.1 interface. The main editor window displays the code for SessionBean1.java. The code includes a private array declaration, a getter method, and a setter method. The setter method is currently selected.

```
130  
131     private Option tytuly [] = new Option[0];  
132     public Option[] getTytuly_ () {  
133         return tytuly_;  
134     }  
135     public void setTytuly_ (Option[] tytuly_) {  
136         this.tytuly_ = tytuly_;  
137     }
```

The status bar at the bottom indicates the cursor is at line 135, column 1, in Insert (INS) mode.

Widok komponentów do realizacji przypadku użycia „dodaj tytuł” (4)

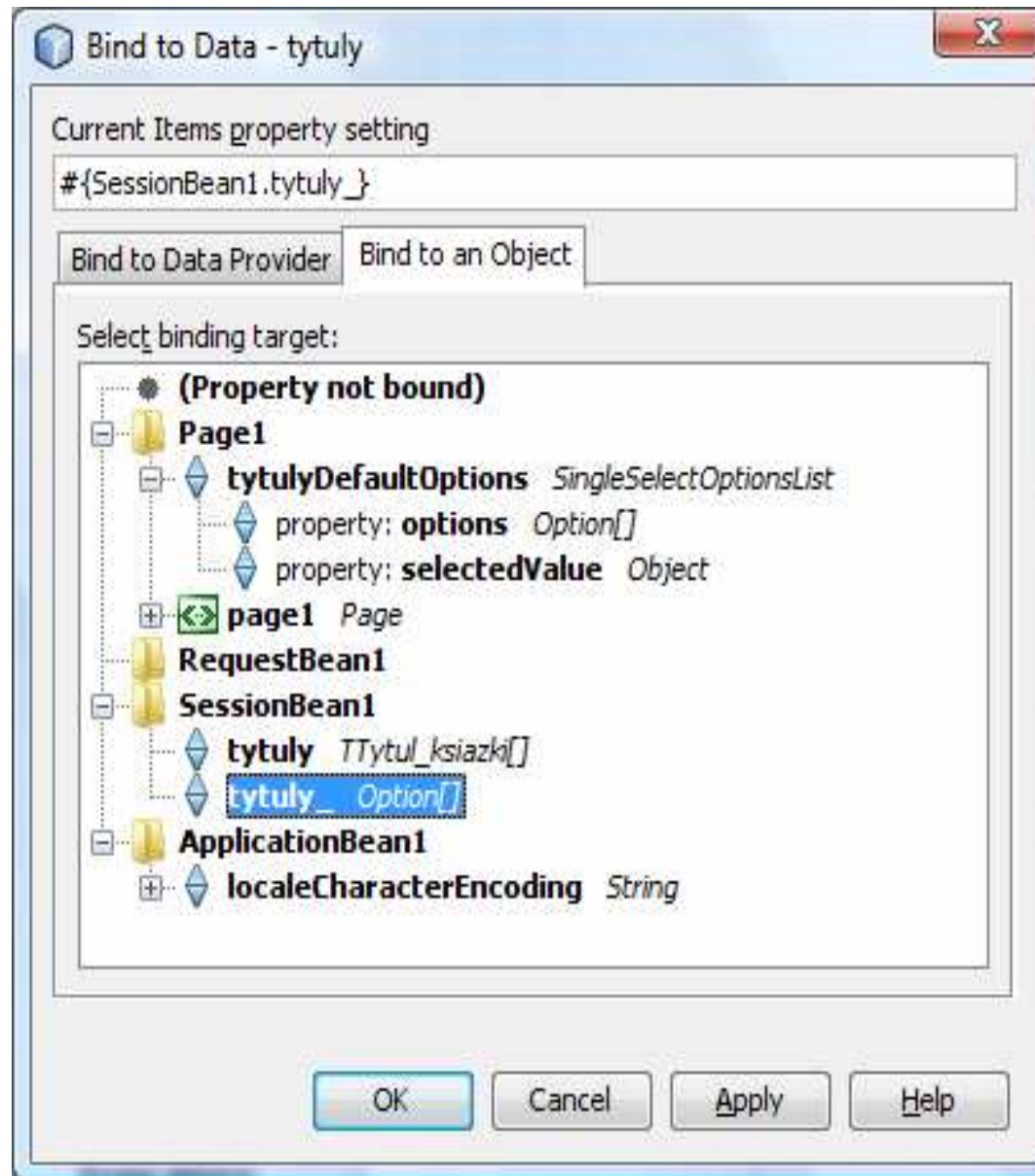
The screenshot displays the NetBeans IDE interface for a web application. The main design view shows a page titled "Tytuły" containing a table with columns: id, tytuł, autor, ISBN, wydawnictwo, and aktor. Below the table are two buttons: "Zapisz tytuły do bazy" and "Zapisz egzemplarze do bazy". The form below has labels and input fields for "Podaj tytuł", "Podaj autora", "Podaj ISBN", "Podaj wydawnictwo", and "Podaj aktora", along with a "Dodaj tytuł" button.

The left-hand "tytuły - Navigator" pane shows the component tree. Red boxes with arrows point to specific components and their annotations:

- id**: Points to the "tytuł" component in the form.
- text**: Points to the "label1:Podaj tytuł" component.
- id**: Points to the "aktor" component in the form.
- text**: Points to the "label5:Podaj aktora" component.
- id**: Points to the "tytuł" component in the table.
- text**: Points to the "Dodaj tytuł" button.

The right-hand "tytuły - Properties" pane shows the properties for the selected component, including "id" (Page1), "Page Layout" (Grid Layout), and "Response Encoding" (UTF-8).

Operacja **Bind to Data** dla komponentu **tytuly** typu **DropDown List (5)**



Tworzenie kodu do połączenia warstwy prezentacji z warstwą biznesową za pomocą referencji do klasy typu Fasada –TAplikacja (6)

The screenshot shows an IDE window with the following structure:

- Projects:** WebWypożyczalnia1, Wypożyczalnia1app
- Source Packages:** webwypożyczalnia1, META-INF, wypożyczalnia1app
- Files:** ApplicationBean1.java, Bundle.properties, Page1.java, RequestBean1.java, SessionBean1.java, TApplikacja.java, TEgzemplarz.java, TEgzemplarzController.java, TEgzemplarz_termin.java

The main editor displays the code for `TAplikacja.java` with the following content:

```
87     private TTytul_ksiazki tytuly[];  
88  
89     public TTytul_ksiazki[] getTytuly() {  
90         return tytuly;  
91     }  
92  
93     public void setTytuly(TTytul_ksiazki[] tytuly) {  
94         this.tytuly = tytuly;  
95     }  
96  
97     public void updateTytuls() {  
98         TTytul_ksiazkiController tytulController =  
99             new TTytul_ksiazkiController();  
100         tytuly = tytulController.getTTytul_ksiazkis();  
101     }  
102     private TApplikacja aplikacja = new TApplikacja();  
103  
104     public TApplikacja getApplikacja() {  
105         return aplikacja;  
106     }  
107  
108     public void setApplikacja(TApplikacja aplikacja) {  
109         this.aplikacja = aplikacja;  
110     }
```

A red rectangular box highlights the following code block:

```
102     private TApplikacja aplikacja = new TApplikacja();  
103  
104     public TApplikacja getApplikacja() {  
105         return aplikacja;  
106     }  
107  
108     public void setApplikacja(TApplikacja aplikacja) {  
109         this.aplikacja = aplikacja;  
110     }
```


Obsługa zdarzenia klikania w przycisk „dodajtytul” w pliku **Page1.java** – realizacja przypadku użycia „dodaj tytul” w warstwie prezentacji (8)

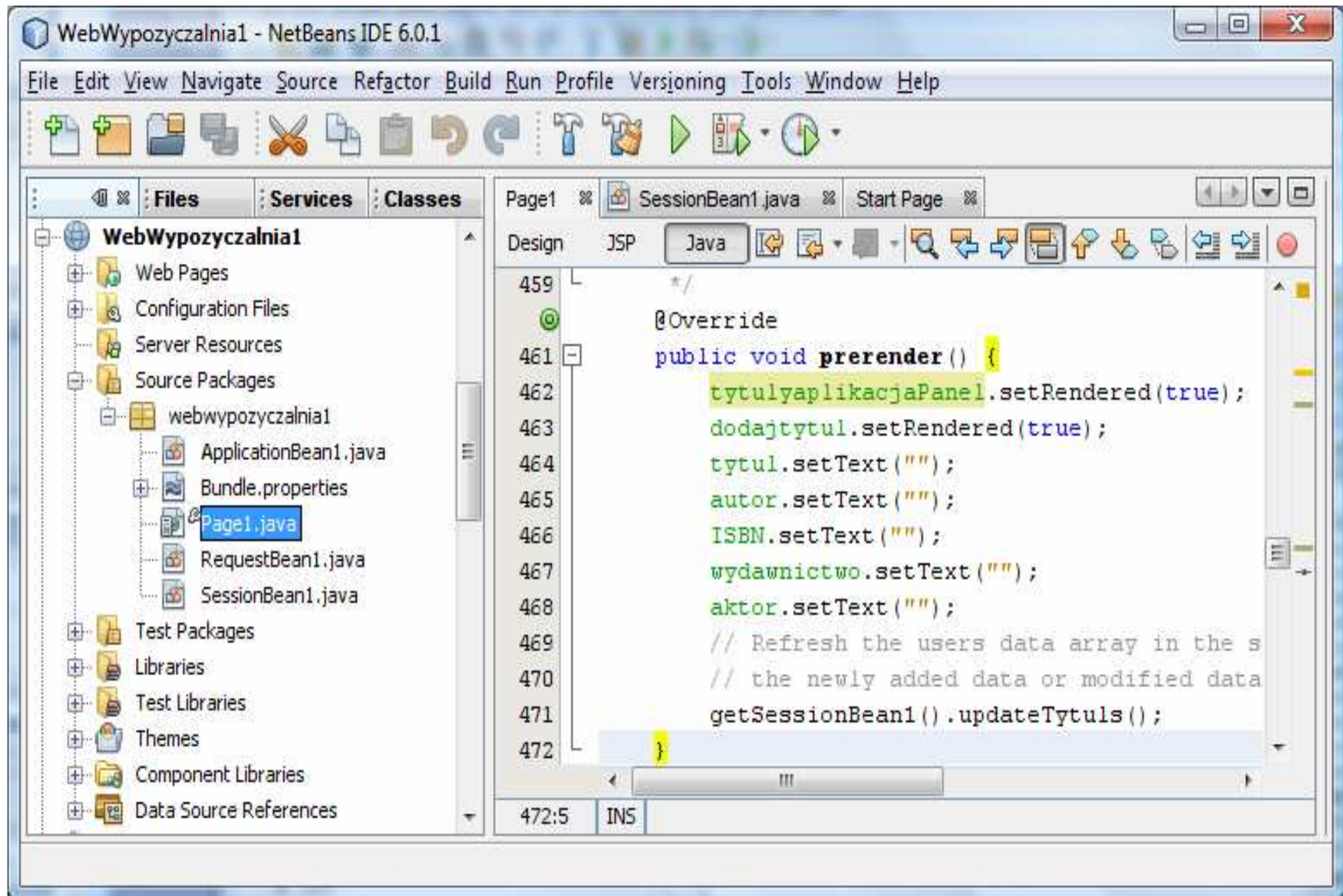
The screenshot displays an IDE window with the following components:

- Project Explorer:** Shows a project named "WebWypożyczalnia1" with a package "webwypożyczalnia1" containing files like "Page1.java" and "SessionBean1.java".
- Code Editor:** Shows the implementation of the `dodajtytul_action()` method in `Page1.java`. The code is as follows:

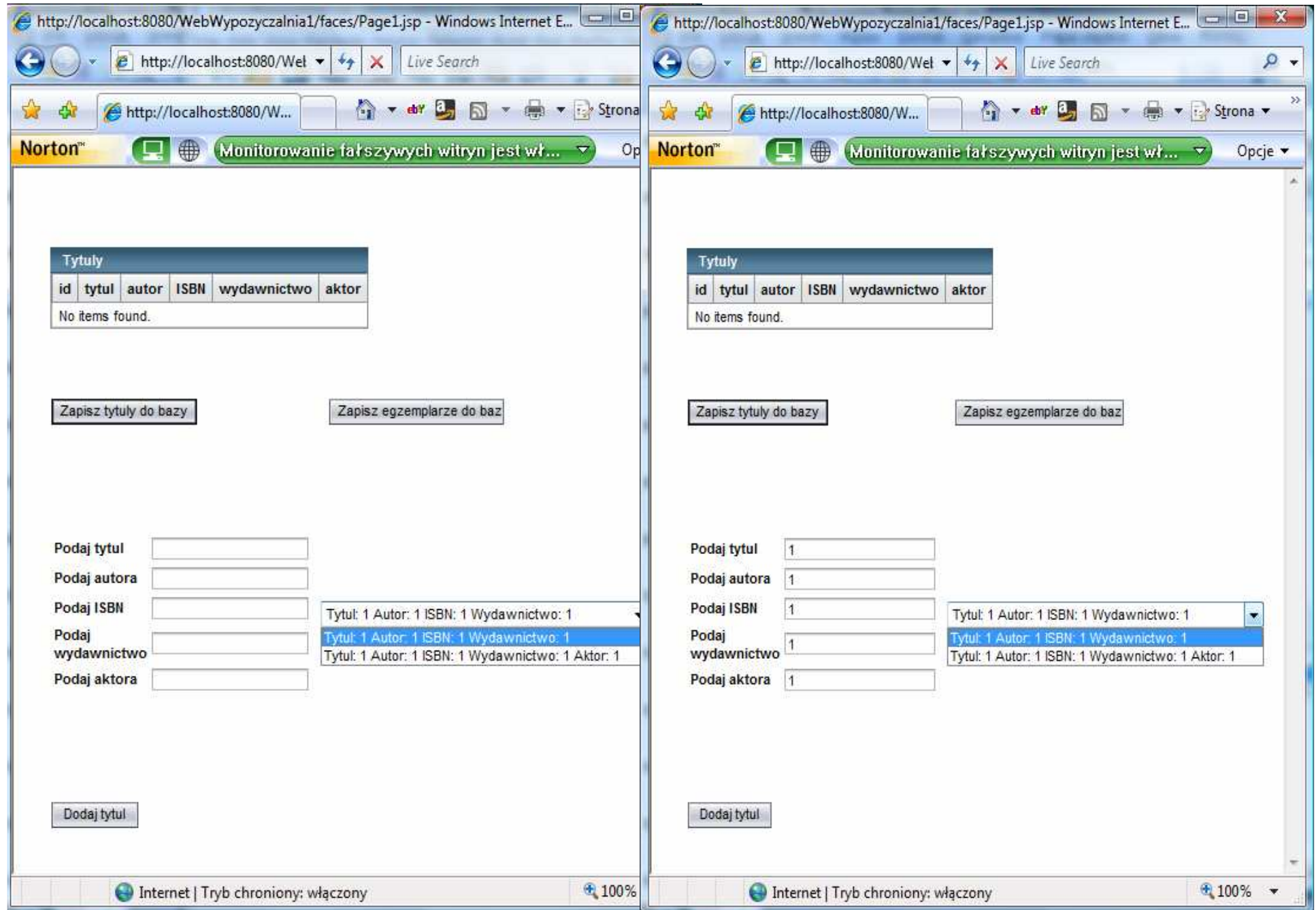
```
512
513 public String dodajtytul_action() {
514     // TODO: Process the action. Return value is a navigation
515     // case name where null will return to the same page.
516     String jaki;
517     if (aktor.getText().equals("")) {
518         jaki = "1";
519     } else {
520         jaki = "3";
521     }
522     String dane[] = {jaki, (String) autor.getText(),
523                     (String) tytul.getText(), (String) ISBN.getText(),
524                     (String) wydawnictwo.getText(), (String) aktor.getText()};
525     getSessionBean1().getAplikacja().dodaj_tytul(dane); // przypadek
526     getSessionBean1().przygotujtytul(); // wyświetlenie kolekcji tyt
527     return null;
528 }
```
- Annotations:** Four red boxes with black text are placed in the code:
 - `//1` is next to the `if` statement (lines 517-521).
 - `//2` is at the end of the method (line 513).
 - `//3` is above the `getSessionBean1().przygotujtytul();` line (line 526).
 - `//4` is above the `getSessionBean1().getAplikacja().dodaj_tytul(dane);` line (line 525).
- Output Window:** Shows the following text:

```
run-display-browser:
run:
BUILD SUCCESSFUL (total time: 3 seconds)
```
- HTTP Monitor:** Shows "WebWypożyczalnia1 (run)".

Ustawienie komponentów w metodzie **prerender (9)** – jest wywoływana w fazie **Response (postback)** np. po zapisaniu danych do bazy danych



Działanie przypadku użycia po stronie aplikacji, bez użycia bazy danych (10)



Tworzenie kodu do zapisu danych o tytułach w bazie danych (11)

```
101 public void zapisz_tytuly_do_bazy() {
102     Ttytul_książki newTtytul_książki;
103     // Add the new Entity to the database using UserController
104     Iterator it = getAplikacja().getTytul_książki().iterator();
105     Ttytul_książkiController Tytul_książkiController =
106         new Ttytul_książkiController();
107     while (it.hasNext()) {
108         newTtytul_książki = (Ttytul_książki) it.next();
109         if (!Tytul_książkiController.findTtytul_książkis(newTtytul_książki))
110             Tytul_książkiController.addTtytul_książkis(newTtytul_książki);
111     }
112 }
```

The screenshot shows the NetBeans IDE interface. The main editor window displays the code for the `zapisz_tytuly_do_bazy()` method. A red box highlights the `SessionBean1.java` tab. Two annotations, `//1` and `//2`, are placed on the right side of the code. `//1` points to the `newTtytul_książki = (Ttytul_książki) it.next();` line, and `//2` points to the `Tytul_książkiController.addTtytul_książkis(newTtytul_książki);` line. The status bar at the bottom shows the cursor is at line 111, column 10.

```
530 public String dodajtytulbaza_action() {
531     // TODO: Replace with your code
532     getSessionBean1().zapisz_tytuly_do_bazy();
533     return null;
534 }
```

The screenshot shows a smaller window of the NetBeans IDE. The main editor window displays the code for the `dodajtytulbaza_action()` method. A red box highlights the `Page1` tab. The code calls the `zapisz_tytuly_do_bazy()` method from the `SessionBean1` object. The status bar at the bottom shows the cursor is at line 532, column 51.

Nowa metoda w klasie TTytul_książkiController do sprawdzania, czy obiekt TTytul_książki w relacji wiele do jeden zostanie zapisany ponownie naruszając więzy integralności referencyjnej tabeli TTytul_książki (12)

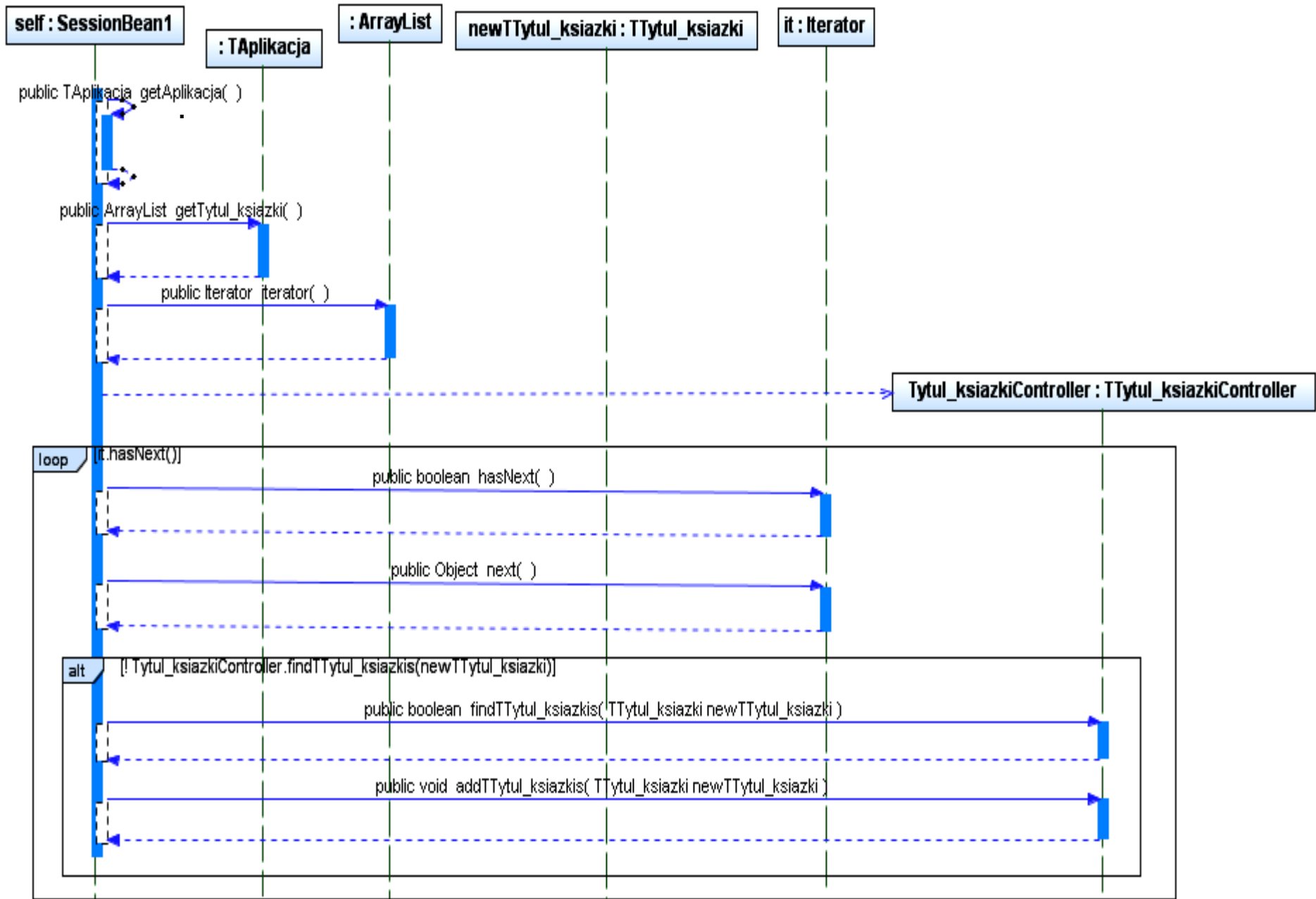
```
public boolean findTTytul_książkis(TTytul_książki TTytul_książki)
{
    EntityManager em = getEntityManager();
    try {
        return em.find(TTytul_książki.class, TTytul_książki.getId())!=null;
    }
    finally {
        em.close();
    }
}
```

Uzupełniono kod konstruktora klasy TTytul_książki– wstawia klucz główny równy 0 do danej, która jeszcze nie została zapisana do bazy danych. Wartość 0 klucza głównego jest wartością nie należącą do dziedziny wartości kluczy głównych-wtedy można przekazać taki obiekt do metody find w klasie EntityMaqnager

```
public TTytul_książki ()
    { id = new Long(-1); //id=null; - wersja 2 w lab7_8
    }
```

Zapis danych do bazy tytułów metodą add obiektu Ttytul_książkiController (13)

public void SessionBean1::zapisz_tytul_do_bazy()



Uruchomienie aplikacji (14)

The screenshot shows a web browser window displaying a web application. The address bar shows the URL `http://localhost:8080/Wel`. The browser's toolbar includes a search bar with the text "Live Search" and a "Strona" menu. A Norton security bar is visible at the top, indicating that "Monitorowanie fałszywych witryn jest wł..." is active.

The main content area features a table titled "Tytuły" with the following data:

id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Below the table are two buttons: "Zapisz tytuły do bazy" and "Zapisz egzemplarze do baz".

At the bottom, there is a form for adding a new title with the following fields:

- Podaj tytuł
- Podaj autora
- Podaj ISBN
- Podaj wydawnictwo
- Podaj aktora

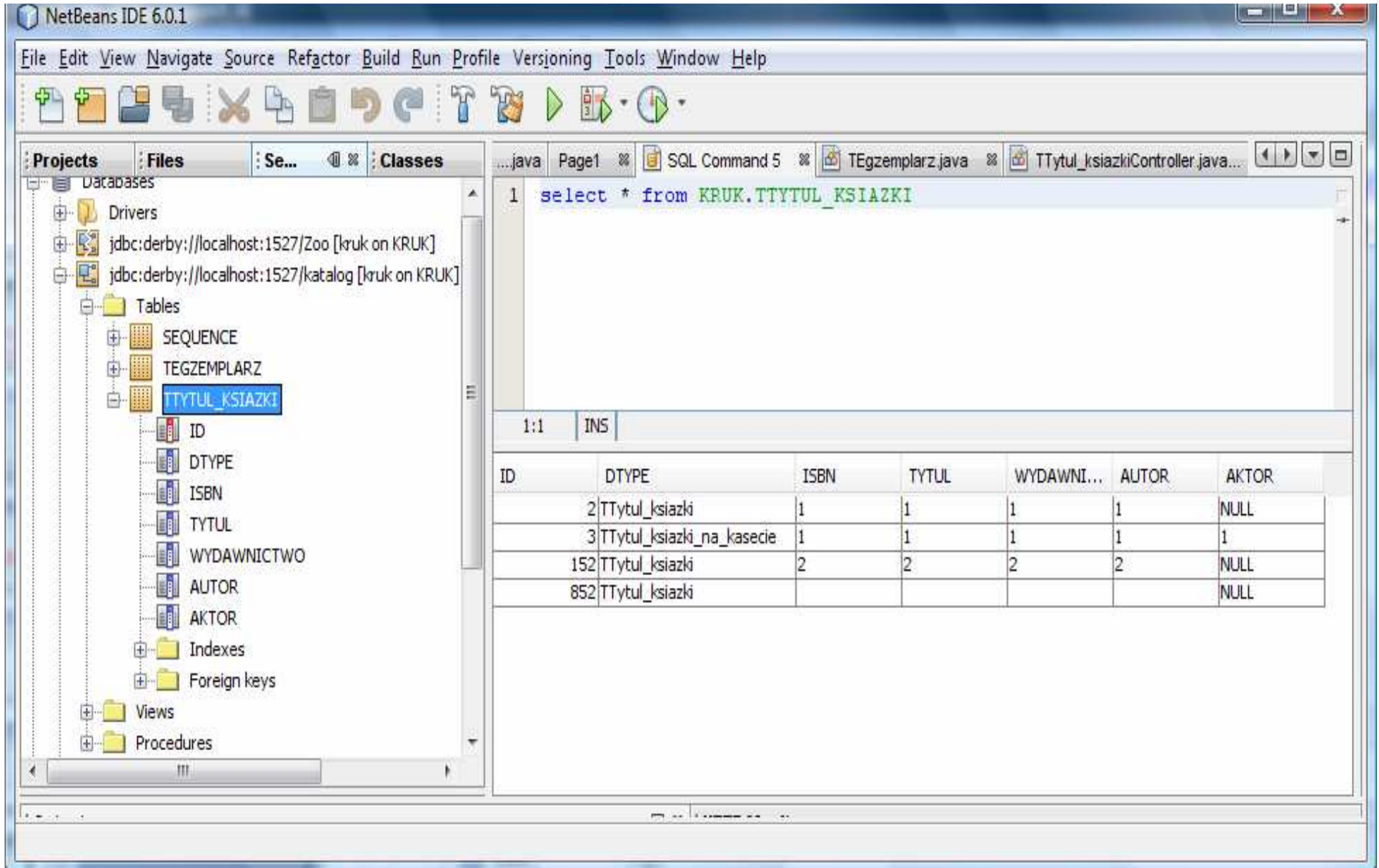
A dropdown menu is open, showing the following options:

- Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
- Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1**
- Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1

A "Dodaj tytuł" button is located at the bottom left of the form area.

The browser's status bar at the bottom indicates "Internet | Tryb chroniony: włączony" and a zoom level of "100%".

Widok bazy danych w zakładce **Services** – wykonanie operacji **ViewData** dostępnej z wyskakującego menu po naciśnięciu tabeli **TTytul_ksiazki** prawym klawiszem myszy (15)



The screenshot shows the NetBeans IDE 6.0.1 interface. The left-hand pane displays the database structure for 'Databases', including 'Drivers', 'jdbc:derby://localhost:1527/Zoo [krak on KRUK]', and 'jdbc:derby://localhost:1527/katalog [krak on KRUK]'. Under 'Tables', the 'TTYTUL_KSIAZKI' table is selected, showing its columns: ID, DTYPE, ISBN, TYTUL, WYDAWNICTWO, AUTOR, and AKTOR. The main window displays the SQL query: `1 select * from KRUK.TTYTUL_KSIAZKI`. Below the query, a data table is shown with the following content:

ID	DTYPE	ISBN	TYTUL	WYDAWNI...	AUTOR	AKTOR
2	TTytul_ksiazki	1	1	1	1	NULL
3	TTytul_ksiazki_na_kasecie	1	1	1	1	1
152	TTytul_ksiazki	2	2	2	2	NULL
852	TTytul_ksiazki					NULL

5. Tworzenie warstwy klienta, prezentacji cd

Integrowanie warstwy prezentacji i klienta
wykonane w technologii **Java Server Faces**

- Wykonanie przypadku użycia „dodaj
egzemplarz”

Tworzenie warstwy prezentacji dla przypadku użycia „dodaj egzemplarz” (1)

- Slajd (1) – uzupełnić komponenty strony **Page1.jsp** wg okienka **Navigator** ze slajdu (1) – **komponenty można wstawiać z palety „przeciągając” je i „upuszczając” w oknie Navigator**. Reprezentuje on zawartość strony **Page1.jsp**. Szczegółowo zawartość każdego z komponentów typu **Grid Panel** pokazano na slajdach: 2,3,4,5. Wg slajdu (2) należy przeciągnąć komponent **Table** do panela **bazaPanel**, nadać **id= bazaksiążki** oraz **title=Książki**. Slajd (3) pokazuje nowy komponent typu **Grid Panel**, który ma **id=książkiaplikacjaPanel**, jego właściwość **columns** ma być ustawiona na 2 i jest zagnieżdżony w panelu **aplikacjadanePanel**. Wg slajdu (4) wstawiono do niego dwie pary komponentów typu **Label** i **Text Field**. Ustawić właściwości **id** oraz **text** tych komponentów w ich okienkach **Properties** wg slajdu (4). Panel **aplikacjadanePanel** ma właściwość **columns** ustawioną na 3. Zgodnie ze slajdem (5) należy przeciągnąć komponent **DropDown List** do panela **aplikacjadane** (zagnieżdżonego w panelu **aplikacjadanePanel**) i nadać mu **id=książki**. Należy przeciągnąć komponent **Button** do panela **aplikacjaoperacjePanel** i nadać mu **id** oraz **text** wg slajdu (5).
- **Należy kliknąć prawym klawiszem myszy na wstawiane komponenty w Okienku Navigator lub Editor i następnie kliknąć na pozycję Add Binding Attribute (punkt 3, slajd 0)**
- Napisać kod wg slajdu (6) w klasie **SessionBean1**. Slajd (7) reprezentuje metodę **updateKsiążkis()**, która umożliwia z pobieranie danych z tabeli **TEgzemplarz** z bazy danych, zapisanych do tablicy **TEgzemplarz książki[]** (slajd(6)) do bindowania z komponentem **bazaksiążki** typu **Table**, Wykonać operację **Bind to Data**, łącząc komponent **bazaksiążki** typu **Table** z tablicą **TEgzemplarz książki[]** - slajd (8) pokazuje stan komponentów po tej operacji. Napisać kod wg slajdu (9) w klasie **SessionBean1**, wpisujący dane do tabeli **Option książki_[]** dla wybranego tytułu książki z listy tytułów książek (**tytuły** typu **DropDown List**) metodą **przygotujksiążki(TTytuł_ książki tytuł)** z warstwy biznesowej. Wykonać operację **Bind to Data**, łącząc komponent **książki** typu **DropDown List** z tablicą **Option książki_[]** – slajd (10). Slajd (11) pokazuje stan komponentów strony po operacji „bindowania”

Tworzenie warstwy prezentacji dla przypadku użycia „dodaj egzemplarz” (2)

- Slajd (12) - wstawianie kodu dla przypadku użycia „dodaj egzemplarz” w warstwie prezentacji. Metoda **dodajksiazke_action()** jako obsługa zdarzenia klikania na przycisk **dodajksiazke** – wybór tytułu następuje w elemencie **tytuly** typu **DropDown List** przez wybór pozycji z listy (**tytuly.getSelected()**), natomiast dane o numerze i ewentualnie terminie podawane są z pól tekstowych **numer** i **termin (12)**. Należy uzupełnić kod metody **prerender** wg slajdu (13). Slajdy (14) i (15) pokazują uruchomioną aplikację.
- Napisać kod w klasie w **SessionBean1** wg slajdu (16). Kod metody **updateAplikacja()** służy do zapisu danych o tytułach i książkach w warstwie biznesowej aplikacji pobierając dane z bazy danych za pomocą tablic **TTytul_książki tytuly[]** oraz **Egzemplarz książki []**. Slajd (17) pokazuje diagram sekwencji metody aktualizującej zawartość warstwy biznesowej.
- Zgodnie ze slajdem (18) napisać w klasie **SessionBean1** metodę **zapisz_książki_do_bazy()** zapisujący dane **zapisz_książki_do_bazy()** aplikacji typu **TEgzemplarz** do bazy danych metodą **addTEgzemplarzs()** klasy **TEgzemplarzController**. Metoda jest wykorzystana do obsługi zdarzenia klikania w przyciski **dodajksiazkabaza (metoda dodajksiazkabaza_action)** w klasie **Page1**. Na slajdzie (20) pokazano diagram sekwencji tej operacji. Slajd (19) przedstawia metodę **findTEgzemplarzs** w klasie **TEgzemplarzController** do sprawdzenia, czy nie ma próby wstawienia ponownie tej samej książki do tabeli **TEgzemplarz** naruszając więzy integralności referencyjnej.
- Na slajdzie (21) pokazano działającą aplikację, która również zapisuje dane o egzemplarzach w aplikacji
- Na slajdach 22-24 pokazano działające przypadki użycia.
- Na slajdzie (25) pokazano zawartość danych bezpośrednio w tabeli **TTytul_książki** w bazie danych (zakładka **Service** i opcja **View Data** dla tabeli)

Przygotowanie komponentów dla przypadku użycia „dodaj egzemplarz”(1)

The screenshot displays the NetBeans IDE 6.0.1 environment. The Project Navigator on the left shows a project structure for 'WebWypożyczalnia1'. The Design view on the right shows a web page layout with two tables: 'Tytuły' and 'Książki'. Below the tables are buttons for 'Zapisz tytuły do bazy', 'Zapisz egzemplarze do bazy', 'Dodaj tytuł', and 'Dodaj egzemplarz'. The Project Navigator shows a hierarchy of components including 'bazaPanel', 'bazaoperacjePanel', and 'aplikacjaoperacjePanel'.

Tytuły

id	tytuł	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

Książki

id	numer	termin	tytuł_książki
123	123	Sun Apr 20 01:32:28 CEST 2008	Tytuł: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Sun Apr 20 01:32:28 CEST 2008	Tytuł: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Sun Apr 20 01:32:28 CEST 2008	Tytuł: abc Autor: abc ISBN: abc Wydawnictwo: abc

Zapisz tytuły do bazy Zapisz egzemplarze do bazy

Podaj tytuł Podaj numer

Podaj autora Podaj termin

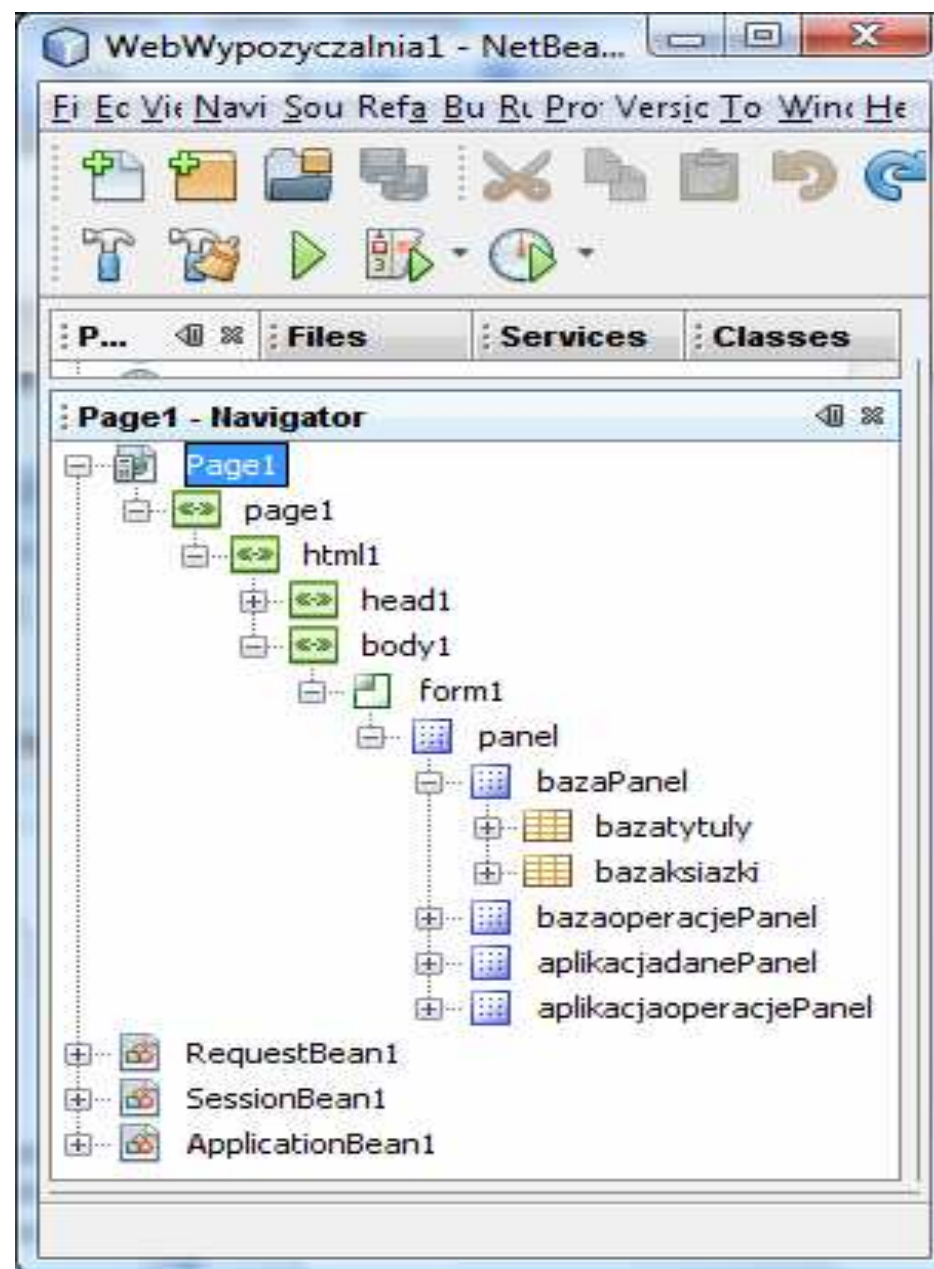
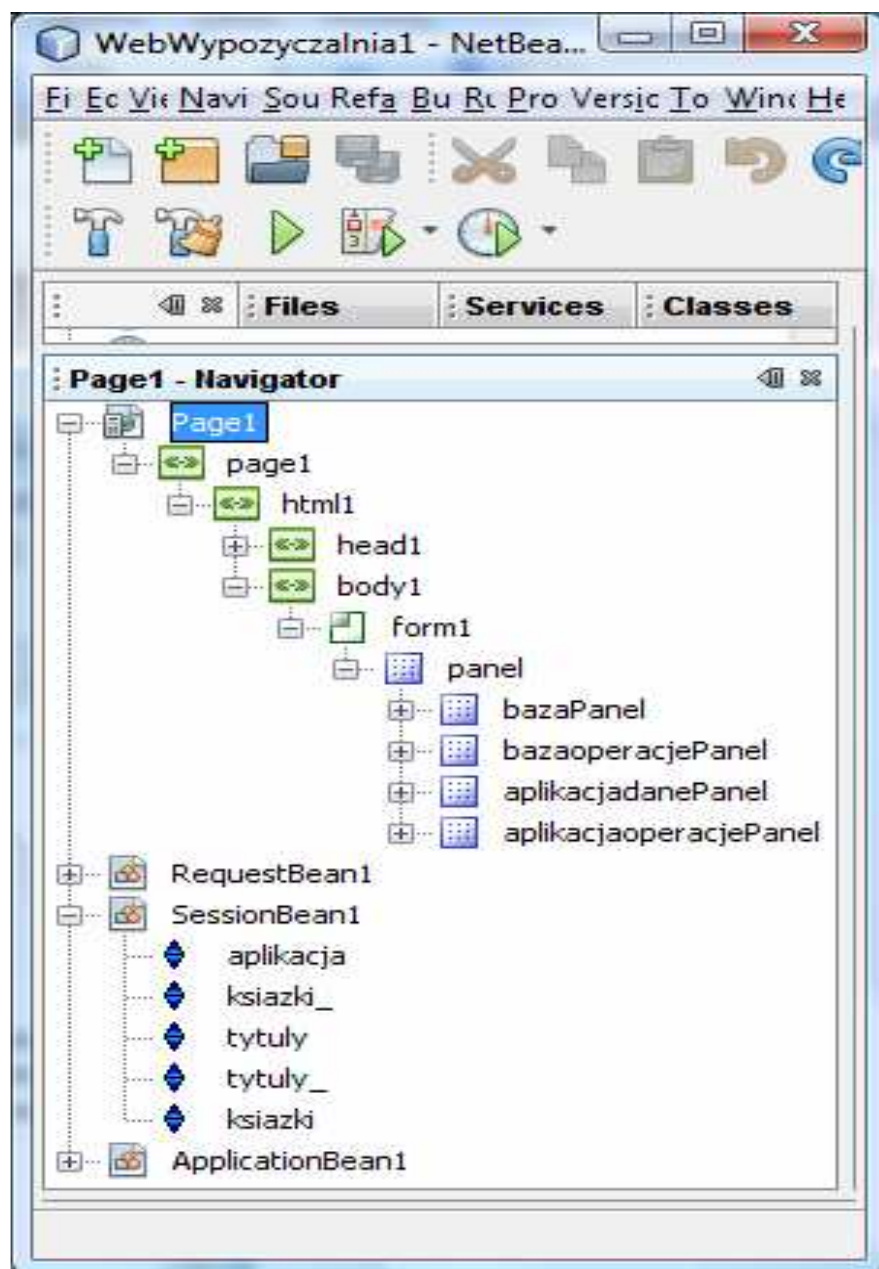
Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Dodaj tytuł Dodaj egzemplarz

Zawartość formularza strony Page1.jsp – przeciągnąć do panela bazaPanel komponent Table – nadać mu id=bazaksiazki (2)



Zawartość formularza strony Page1.jsp (3)

The image displays two side-by-side screenshots of the NetBeans IDE 6.0.1 interface, showing the 'Page1 - Navigator' window. Both screenshots show a tree view of the page structure, including components like 'Page1', 'page1', 'html1', 'head1', 'body1', 'form1', 'panel', 'bazaPanel', 'bazaoperacjePanel', 'aplikacjadanePanel', and 'aplikacjaoperacjePanel'. The left screenshot shows the tree view with the following structure:

- Page1
 - page1
 - html1
 - head1
 - body1
 - form1
 - panel
 - bazaPanel
 - bazaoperacjePanel
 - dodajtytulbaza:Zapisz tytuly do bazy
 - dodajksiazkabaza:Zapisz egzemplarze do bazy
 - aplikacjadanePanel
 - aplikacjaoperacjePanel
- RequestBean1
- SessionBean1
- ApplicationBean1

The right screenshot shows the same tree view, but with an additional component, 'aplikacjadanePanel', added under the 'panel' node:

- Page1
 - page1
 - html1
 - head1
 - body1
 - form1
 - panel
 - bazaPanel
 - bazaoperacjePanel
 - aplikacjadanePanel
 - tytulyaplikacjaPanel
 - ksiazkiaplikacjadane
 - aplikacjadane
 - aplikacjaoperacjePanel
- RequestBean1
- SessionBean1
- ApplicationBean1

Zawartość formularza strony Page1.jsp (4)

WebWypożyczalnia1 - NetBeans IDE 6.0.1

Page1 - Navigator

- Page1
 - page1
 - html1
 - head1
 - body1
 - Form1
 - panel
 - bazaPanel
 - bazaoperacjePanel
 - aplikacjadanePanel
 - tytułyaplikacjaPanel
 - label1:Podaj tytuł
 - tytuł
 - label2:Podaj autora
 - autor
 - label3:Podaj ISBN
 - ISBN
 - label4:Podaj wydawnictwo
 - wydawnictwo
 - label5:Podaj aktora
 - aktor
 - ksiazkiaplikacjadane
 - aplikacjadane
 - aplikacjaoperacjePanel

RequestBean1
SessionBean1
ApplicationBean1

WebWypożyczalnia1 - NetBeans IDE 6.0.1

Page1 - Navigator

- Page1
 - page1
 - html1
 - head1
 - body1
 - form1
 - panel
 - bazaPanel
 - bazaoperacjePanel
 - aplikacjadanePanel
 - tytułyaplikacjaPanel
 - ksiazkiaplikacjadane
 - label6:Podaj numer
 - numer
 - label7:Podaj termin
 - termin
 - aplikacjadane
 - aplikacjaoperacjePanel

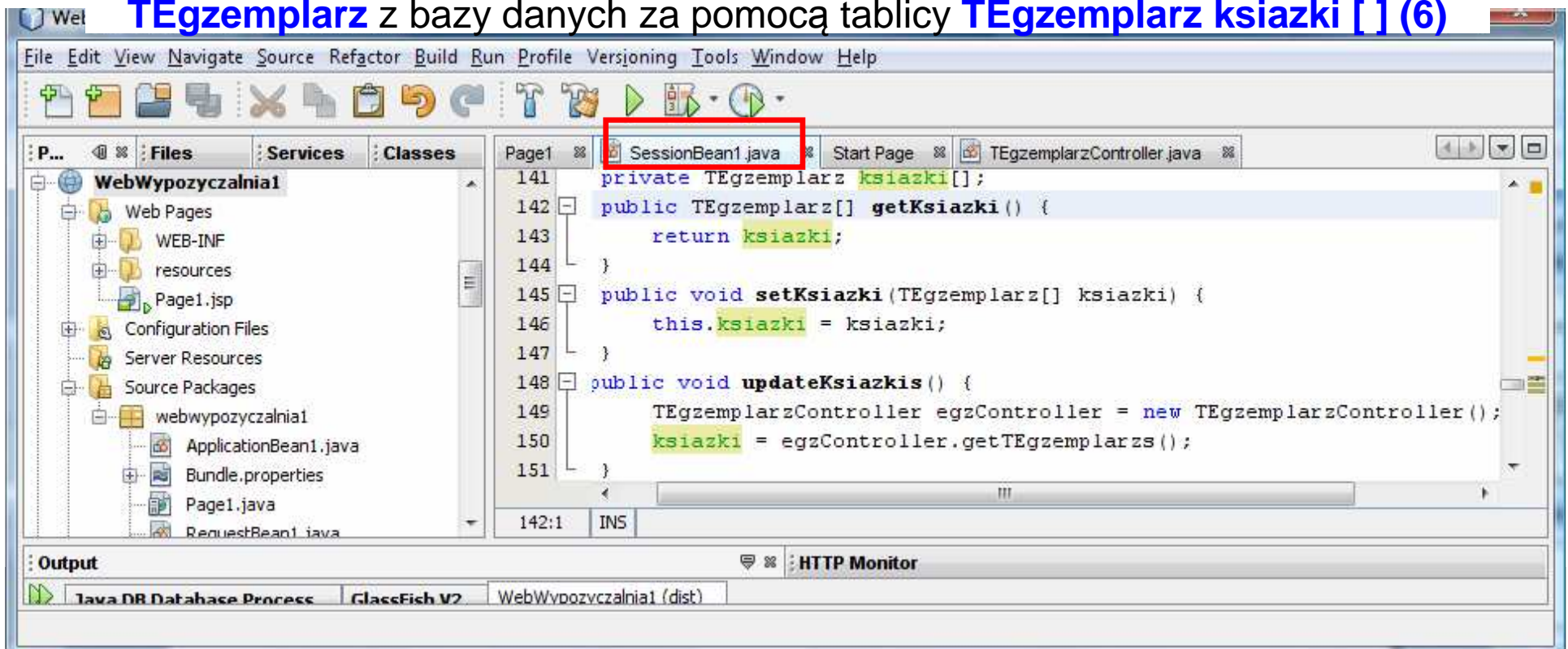
RequestBean1
SessionBean1
ApplicationBean1

Annotations: text, id, id, text

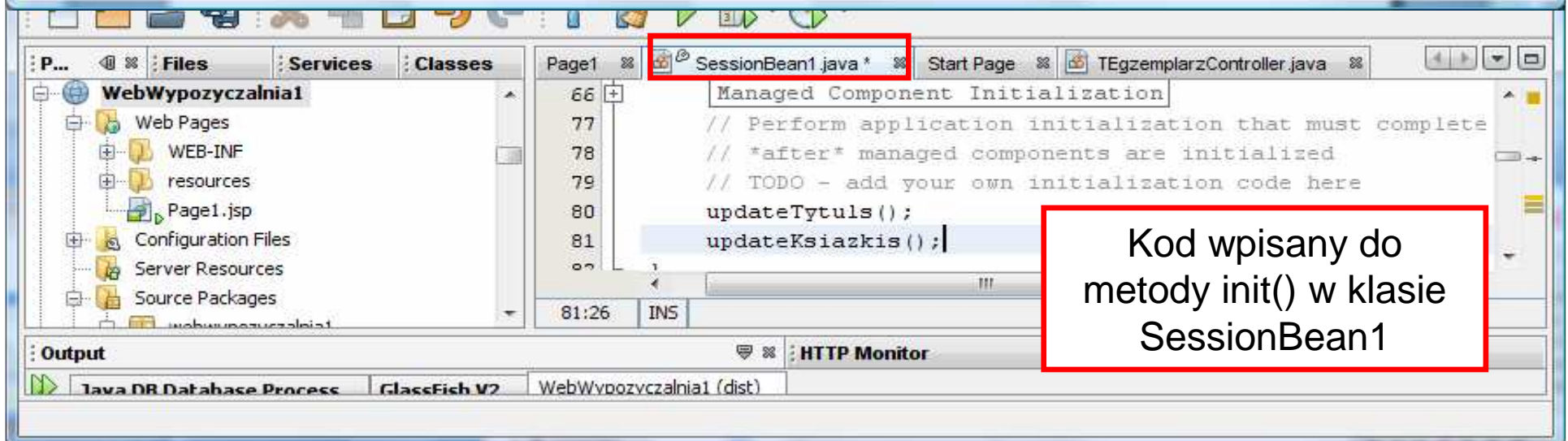
Zawartość formularza strony Page1.jsp (5)

The image displays two side-by-side screenshots of the NetBeans IDE interface, both showing the 'Page1 - Navigator' window. The left window shows the full project structure, including 'Page1', 'page1', 'html1', 'head1', 'body1', 'form1', 'panel', 'bazaPanel', 'bazaoperacjePanel', 'aplikacijadanePanel', 'tytulyaplikacjaPanel', 'ksiazkiaplikacijadane', 'aplikacijadane', 'tytuly', 'ksiazki', and 'aplikacjaoperacjePanel'. The right window shows the same structure but highlights specific components with red boxes and arrows. The components highlighted are 'id' and 'text' in red boxes, with arrows pointing to 'aplikacjaoperacjePanel' and 'dodajtytul: Dodaj tytul' and 'dodajksiazke: Dodaj egzemplarz'. The 'id' and 'text' labels are also shown in red boxes at the bottom of the right window, with arrows pointing to the 'dodajtytul: Dodaj tytul' and 'dodajksiazke: Dodaj egzemplarz' components respectively.

Tworzenie kodu do „bindowania” komponentu **Table** z danymi z tabeli **TEgzemplarz** z bazy danych za pomocą tablicy **TEgzemplarz ksiazki [] (6)**



```
141 private TEgzemplarz ksiazki[];
142 public TEgzemplarz[] getKsiazki() {
143     return ksiazki;
144 }
145 public void setKsiazki(TEgzemplarz[] ksiazki) {
146     this.ksiazki = ksiazki;
147 }
148 public void updateKsiazkis() {
149     TEgzemplarzController egzController = new TEgzemplarzController();
150     ksiazki = egzController.getTEgzemplarzs();
151 }
```

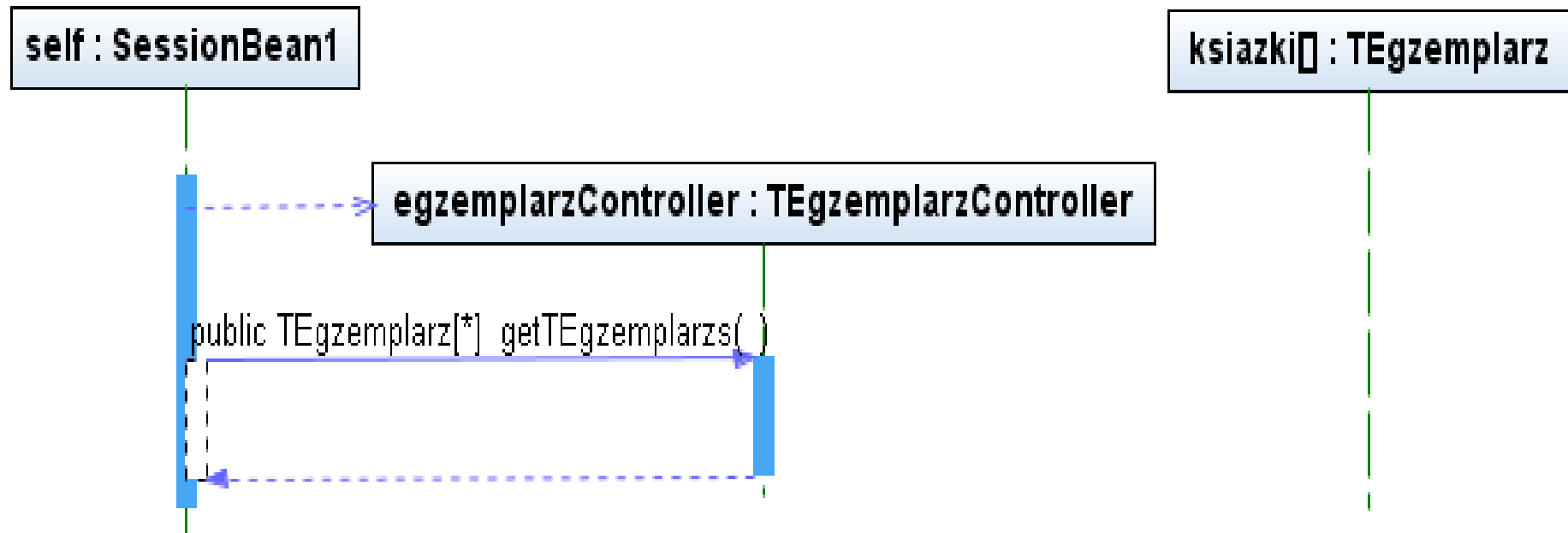


```
66 Managed Component Initialization
77 // Perform application initialization that must complete
78 // *after* managed components are initialized
79 // TODO - add your own initialization code here
80 updateTytuls();
81 updateKsiazkis();
```

Kod wpisany do metody init() w klasie SessionBean1

Odczyt książek z bazy danych (7)

public void SessionBean1::updateKsiazkis()



Wstawianie nowych komponentów rozpinanych na zagnieżdżonych komponentach typu **Layout – Grid Panel (8)**

The screenshot shows an IDE window titled "Web" with a menu bar (File, Edit, View, Navigate, Source, Refactor, Build, Run, Profile, Versioning, Tools, Window, Help) and a toolbar. The main workspace displays a design view with two tables and form elements.

Tytuły

id	tytul	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

Książki

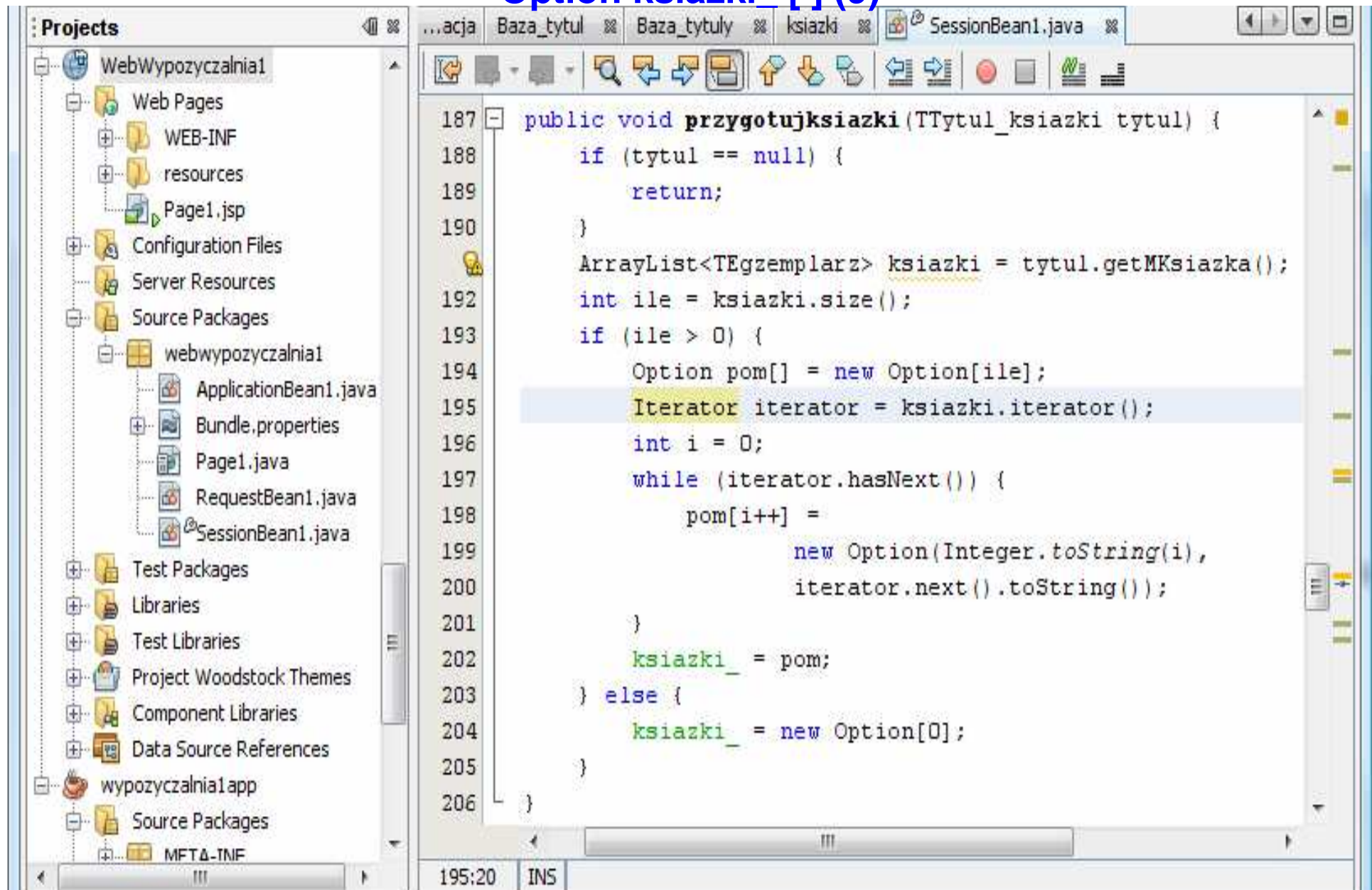
id	numer	termin	tytul_ksiazki
123	123	Thu Apr 17 13:16:29 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Thu Apr 17 13:16:29 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Thu Apr 17 13:16:29 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc

Formy:

- Zapisz tytuły do bazy
- Zapisz egzemplarze do bazy
- Podaj tytuł
- Podaj autora
- Podaj ISBN
- Podaj wydawnictwo
- Podaj aktora
- Podaj numer
- Podaj termin
- Dodaj tytuł
- Dodaj egzemplarz

Output: GlassFish V2 | Java DB Database Process | WebWypożyczalnia1 (run)

Wstawienie kodu do wyświetlania danych o egzemplarzach przechowywanych w aplikacji w elemencie **ksiazki** typu **DropDown List** za pośrednictwem tablicy **Option ksiazki_ [] (9)**



The screenshot shows an IDE window with a project explorer on the left and a code editor on the right. The project explorer shows a project named 'WebWypożyczalnia1' with a package 'webwypożyczalnia1' containing several Java files, including 'SessionBean1.java'. The code editor displays the following Java code:

```
187 public void przygotujksiazki (TTytul_ksiazki tytul) {
188     if (tytul == null) {
189         return;
190     }
191     ArrayList<TEgzemplarz> ksiazki = tytul.getMKsiazka();
192     int ile = ksiazki.size();
193     if (ile > 0) {
194         Option pom[] = new Option[ile];
195         Iterator iterator = ksiazki.iterator();
196         int i = 0;
197         while (iterator.hasNext()) {
198             pom[i++] =
199                 new Option(Integer.toString(i),
200                             iterator.next().toString());
201         }
202         ksiazki_ = pom;
203     } else {
204         ksiazki_ = new Option[0];
205     }
206 }
```

The line `Iterator iterator = ksiazki.iterator();` at line 195 is highlighted in blue. The status bar at the bottom shows '195:20 INS'.

Wykonanie operacji **Bind to Data** między elementem **ksiązki** typu **DropDown List** oraz tablicą **Option książki_ [] (10)**

The screenshot shows an IDE window with a 'Bind to Data - książki' dialog box open. The dialog box has the following content:

Current Items property setting: `#{SessionBean1.ksiązki_}`

Bind to Data Provider: Bind to an Object

Select binding target:

- (Property not bound)
- Page1
 - ksiązkiDefaultOptions SingleSelectOptionsList
 - property: options Option[]
 - property: selectedValue Object
 - page1 Page
 - RequestBean1
 - SessionBean1
 - aplikacja TApplikacja
 - ksiązki Option[]** (highlighted)
 - tytuly TTytul_ksiązki[]
 - tytuly_ Option[]
 - ksiązki TEgzemplarz[]
 - ApplicationBean1
 - localeCharacterEncoding String

Buttons: OK, Cancel, Apply, Help

The background shows a web page design with a table and a dropdown list. The table has columns: ISBN, wydawnictwo, aktor, id, num. The dropdown list is labeled 'a...'. The IDE interface includes a menu bar (File, Edit, View, etc.), a toolbar, and a status bar at the bottom showing 'GlassFish V2', 'Java DB Database Process', and 'WebWypożyczalnia1 (run)'.

Efekt operacji **Bind to Data** – okienko **Navigator** przedstawia strukturę paneli i komponentów (11)

The screenshot shows an IDE window titled 'Web' with a menu bar (File, Edit) and a toolbar. The main workspace is in 'Design' mode, showing a JSP page with two data tables and form elements.

Navigator (Left Panel):

- ksiązki - Navigator
 - form1
 - panel
 - bazaPanel
 - bazatytyly
 - bazaksiążki
 - bazaoperacjePanel
 - dodajtytulbaza:Zapisz tytuly do ba...
 - dodajksiazkabaza:Zapisz egzemplar...
 - aplikacjadanePanel
 - tytulyaplikacjaPanel
 - label1:Podaj tytuly
 - tytul
 - label2:Podaj autora
 - autor
 - label3:Podaj ISBN
 - ISBN
 - label4:Podaj wydawnictwo
 - wydawnictwo
 - label5:Podaj aktora
 - aktor
 - ksiazkiaplikacjadane
 - label6:Podaj numer
 - numer
 - label7:Podaj termin
 - termin
 - aplikacjaoperacjePanel
 - tytul
 - ksiazki

Tytuły					
id	tytul	autor	ISBN	wydawnictwo	aktor
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc
123	abc	abc	abc	abc	abc

Książki			
id	numer	termin	tytul_ksiazki
123	123	Thu Apr 17 13:20:50 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Thu Apr 17 13:20:50 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc
123	123	Thu Apr 17 13:20:50 CEST 2008	Tytul: abc Autor: abc ISBN: abc Wydawnictwo: abc

Zapisz tytuly do bazy

Zapisz egzemplarze do bazy

Podaj tytuly

Podaj autora

Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Podaj numer

Podaj termin

Dodaj tytuly

Dodaj egzemplarz

Wstawianie kodu dla przypadku użycia „**dodaj egzemplarz**” w warstwie prezentacji – wybór tytułu następuje w elemencie **tytuly** typu **DropDown List** przez wybór pozycji z listy (`tytuly.getSelected()`), natomiast dane o numerze i ewentualnie terminie podawane są z pól tekstowych **numer** i **termin** (12)

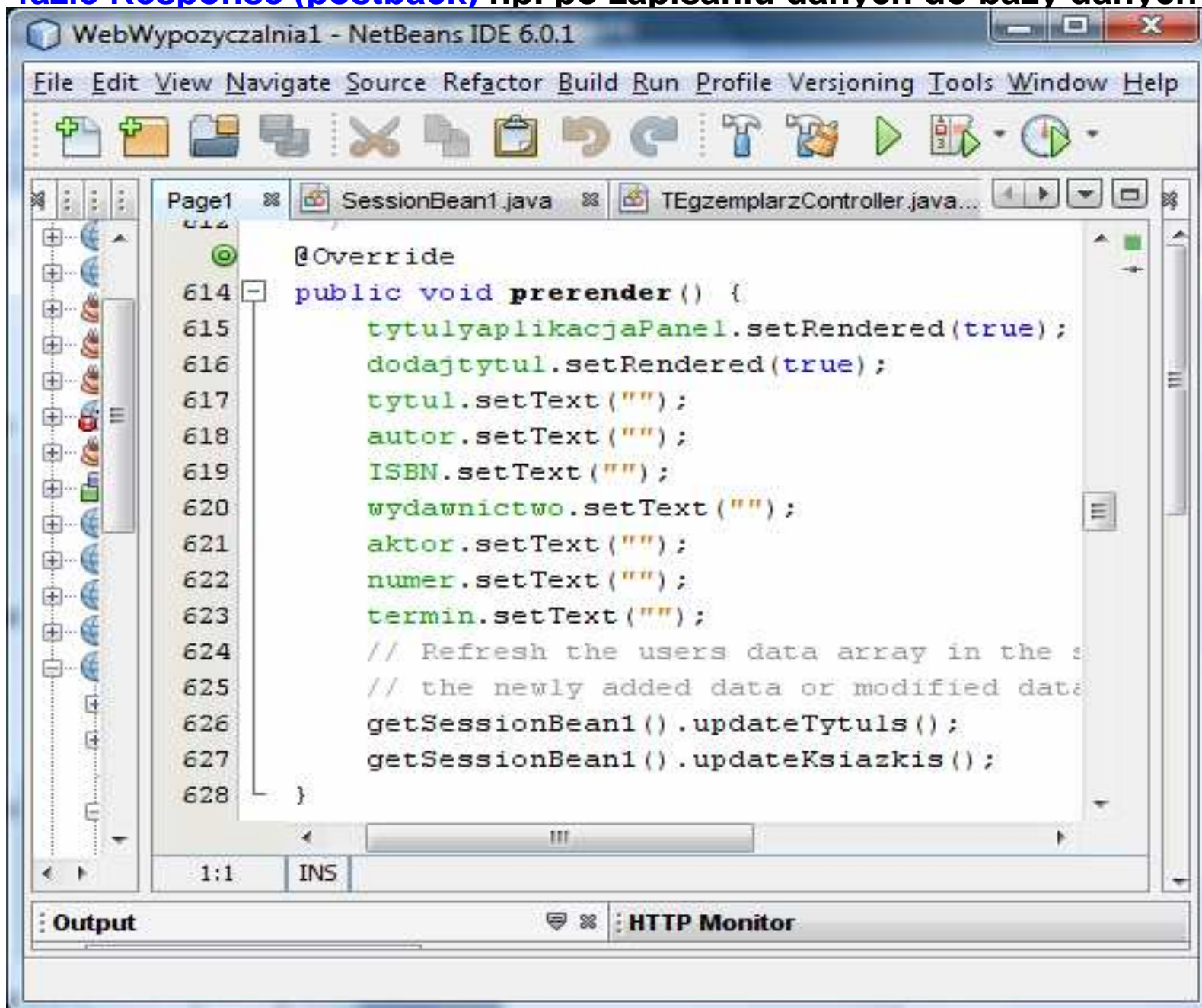
```
...java | SQL Command 6 | TApplikacja.java | Page1 * | ApplicationBean1.java | Start Page |
725 public String dodajksiazke_action() {
726     // TODO: Process the action. Return value is a navigation
727     // case name where null will return to the same page.
728     String jaki,jaka;
729     if (termin.getText().equals(""))
730     { jaka = "0"; }
731     else
732     { jaka = "1"; } //Informacja dla fabryki - jaki tytuł wyszukać
733     int nr = Integer.parseInt((String) tytuly.getSelected());
734     if (nr==0) return null;
735     Option pom[]=getSessionBean1().getTytuly_();
736     String pom1=pom[nr-1].getLabel();
737     String pom2[]=pom1.split(" ");
738     if (pom2.length==8) //Informacja dla fabryki - jaki egzemplarz wykonać
739     { pom1="";
740       jaki="0"; }
741     else
742     { pom1=pom2[9];
743       jaki="2"; }
744     String dane1[] = {jaki, (String) pom2[5], pom1};
745     String dane2[] = {jaka, (String) numer.getText(), (String) termin.getText()};
746     getSessionBean1().przygotujksiazki(
747         getSessionBean1().getApplikacja().dodaj_ksiazke(dane1, dane2));
748     return null;
749 }
```

Wybór typu klasy egzemplarz:
0 – TEgzemplarz
1 – TEgzemplarz_termin

Pobranie z listy rozwijanej typu DropDown informacji o tytule książki, dla którego należy dodać książkę

Wybór typu klasy tytuł do wyszukiwania:
0 – TTytuł_ksiazki (4 dane – 8 łańcuchów w łańcuchu tytułu)
2 – TTytuł_ksiazki_na_kasecie (5 danych – 10 łańcuchów w łańcuchu tytułu)

Uzupełnienie kodu metody **prerender** w klasie **Page1.java (13)** – jest wywoływana w **fazie Response (postback)** np. po zapisaniu danych do bazy danych



The screenshot shows the NetBeans IDE 6.0.1 interface. The main editor window displays the code for the `prerender` method in `Page1.java`. The code is as follows:

```
@Override
614 public void prerender() {
615     tytułaplikacjaPanel.setRendered(true);
616     dodajtytuł.setRendered(true);
617     tytuł.setText("");
618     autor.setText("");
619     ISBN.setText("");
620     wydawnictwo.setText("");
621     aktor.setText("");
622     numer.setText("");
623     termin.setText("");
624     // Refresh the users data array in the s
625     // the newly added data or modified data
626     getSessionBean1().updateTytułs();
627     getSessionBean1().updateKsiazkis();
628 }
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Source, Refactor, Build, Run, Profile, Versioning, Tools, Window, Help), a toolbar with various icons, and a project browser on the left. The status bar at the bottom shows "1:1" and "INS".

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp - Windows Internet Explorer

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp

http://localhost:8080/WebWypożyczalnia1/faces/...

Norton

Monitorowanie fałszywych witryn jest włączone

Opcje

Tytuły					
id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki			
id	numer	termin	tytuł_książki
No items found.			

Zapisz tytuły do bazy

Zapisz egzemplarze do baz

Podaj tytuł

Podaj autora

Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Podaj numer

Podaj termin

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2 Numer: 2 termin: Tue Apr 22 14:52:59 CEST 2008

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2

Tytuł: 3 Autor: 3 ISBN: 3 Wydawnictwo: 3

Dodaj tytuł

Dodaj egzemplarz

Internet | Tryb chroniony: włączony

100%

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp - Windows Internet Explorer

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp

http://localhost:8080/WebWypożyczalnia1/faces/...

Norton

Monitorowanie fałszywych witryn jest włączone

Opcje

Tytuły					
id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki			
id	numer	termin	tytuł_książki
No items found.			

Zapisz tytuły do bazy

Zapisz egzemplarze do baz

Podaj tytuł

Podaj autora

Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Podaj numer

Podaj termin

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2 Numer: 2 termin: Tue Apr 22 14:52:59 CEST 2008

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2 Numer: 2 termin: Tue Apr 22 14:52:59 CEST 2008

Tytuł: 2 Autor: 2 ISBN: 2 Wydawnictwo: 2 Aktor: 2 Numer: 3

Dodaj tytuł

Dodaj egzemplarz

Internet | Tryb chroniony: włączony

100%

Aktualizacja danych w aplikacji pobieranych z bazy danych za pomocą tablic **TTytul_książki tytuly[]** oraz **TEgzemplarz książki []** (16)

WebWypożyczalnia1 - NetBeans IDE 6.5

```
112 private TAplikacja aplikacja = new TAplikacja();
113 public TAplikacja getAplikacja() {
114     return aplikacja;
115 }
116 public void setAplikacja(TAplikacja aplikacja) {
117     this.aplikacja = aplikacja;
118 }
119 public void updateAplikacja() {
120     for (int i = 0; i < tytuly.length; i++)
121         aplikacja.getTytul_ksiazki().add(tytuly[i]);
122     Iterator it = aplikacja.getTytul_ksiazki().iterator();
123     while (it.hasNext()) {
124         TTytul_ksiazki tytul = (TTytul_ksiazki) it.next();
125         for (int j = 0; j < ksiazki.length; j++) {
126             if (ksiazki[j].getMTytul_ksiazki().equals(tytul))
127                 tytul.getMKsiazka().add(ksiazki[j]);
128         }
129     }
130 }
```

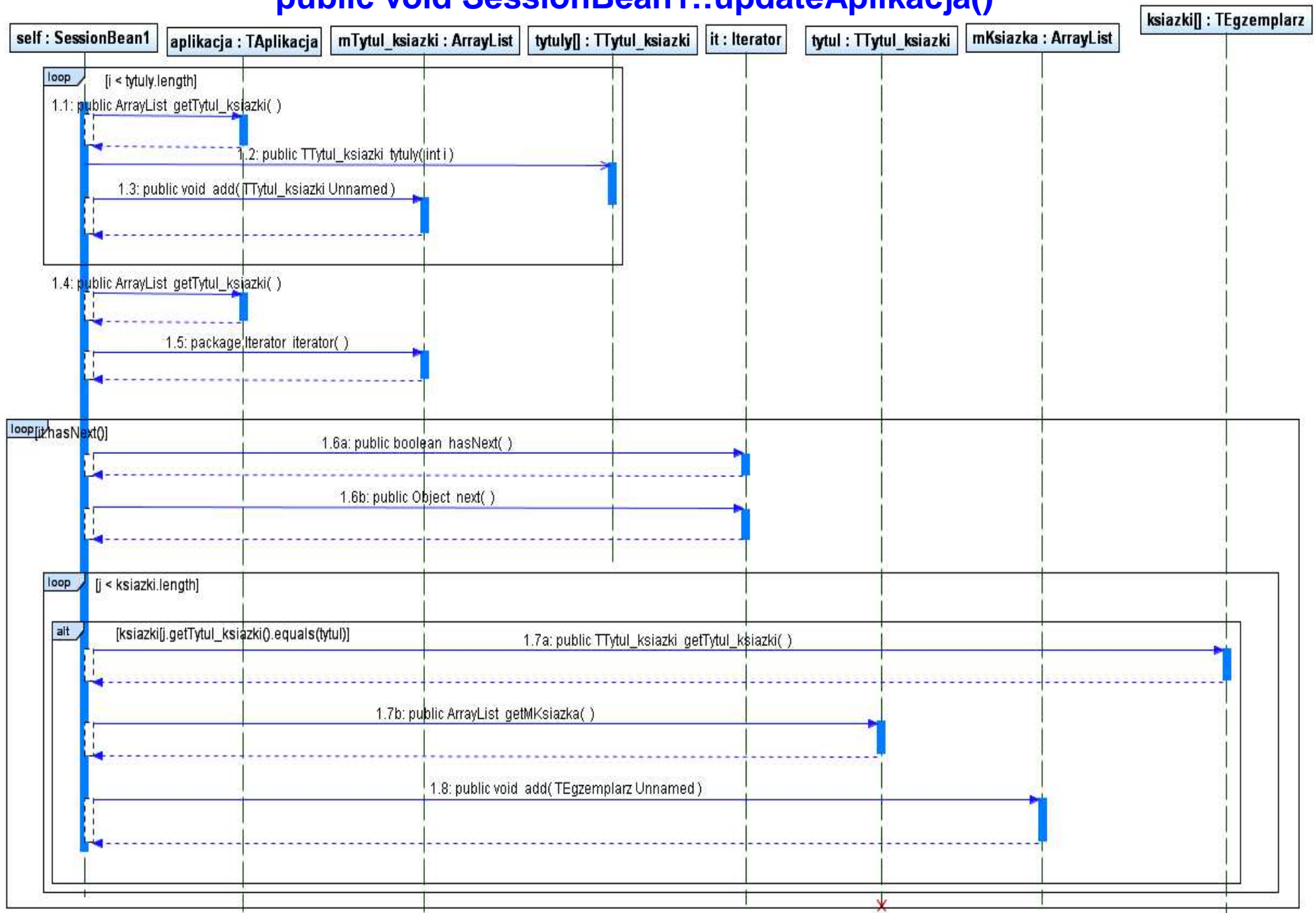
WebWypożyczalnia1 - NetBeans I...

```
@Override
60 public void init() {
61     // Perform initiali
62     super.init();
63     // Perform applicat
64     // *before* managed
65     // TODO - add your
66
67     Managed Component
78     // Perform applicat
79     // *after* managed
80     // TODO - add your
81     updateTytuls();
82     updateKsiazkis();
83     updateAplikacja();
84     przygotujtytuly();
85 }
```

Nazwa metody po zestandaryzowaniu nazw dla atrybutu mTytul_ksiazki w klasie TEgzemplarz
Stąd: getMTytul_ksiazki oraz setMTytul_ksiazki

Odczyt danych z bazy tytułów oraz książek za pomocą tablic encji tytuły i książki (17)

public void SessionBean1::updateAplikacja()



Wykonanie kodu zapisującego dane o egzemplarzach przechowywanych w aplikacji do bazy danych (18)

The image displays two screenshots of the NetBeans IDE 6.0.1 interface, illustrating the implementation of a database saving function.

Top Screenshot: Shows the `SessionBean1.java` file. The method `zapisz_książki_do_bazy()` is defined, which iterates through book titles and their corresponding example IDs to save them to the database. The code is as follows:

```
182 public void zapisz_książki_do_bazy() {
183     TEgzemplarz newTEgzemplarz;
184     TTYtul_książki newTTYtul_książki;
185     // Add the new Entity to the database using UserController
186     Iterator it = getAplikacja().getTytul_książki().iterator();
187     TEgzemplarzController egzemplarzController =
188         new TEgzemplarzController();
189     while (it.hasNext()) {
190         newTTYtul_książki = (TTYtul_książki) it.next();
191         Iterator it_ = newTTYtul_książki.getMKsiążka().iterator();
192         while (it_.hasNext()) {
193             newTEgzemplarz = (TEgzemplarz) it_.next();
194             if (!egzemplarzController.findTEgzemplarz(newTEgzemplarz))
195                 egzemplarzController.addTEgzemplarz(newTEgzemplarz);
196         }
197     }
198 }
```

Bottom Screenshot: Shows the `Page1` file. The method `dodajksiążkabaza_action()` is defined, which calls the `zapisz_książki_do_bazy()` method from the `SessionBean1` class. The code is as follows:

```
692 public String dodajksiążkabaza_action() {
693     // TODO: Replace with your code
694     getSessionBean1().zapisz_książki_do_bazy();
695     return null;
696 }
```

Nowa metoda w klasie TEgzemplarzController do sprawdzania, czy obiekt TEgzemplarz w relacji wiele do jeden zostanie zapisany ponownie naruszając więzy integralności referencyjnej tabeli TEgzemplarz (19)

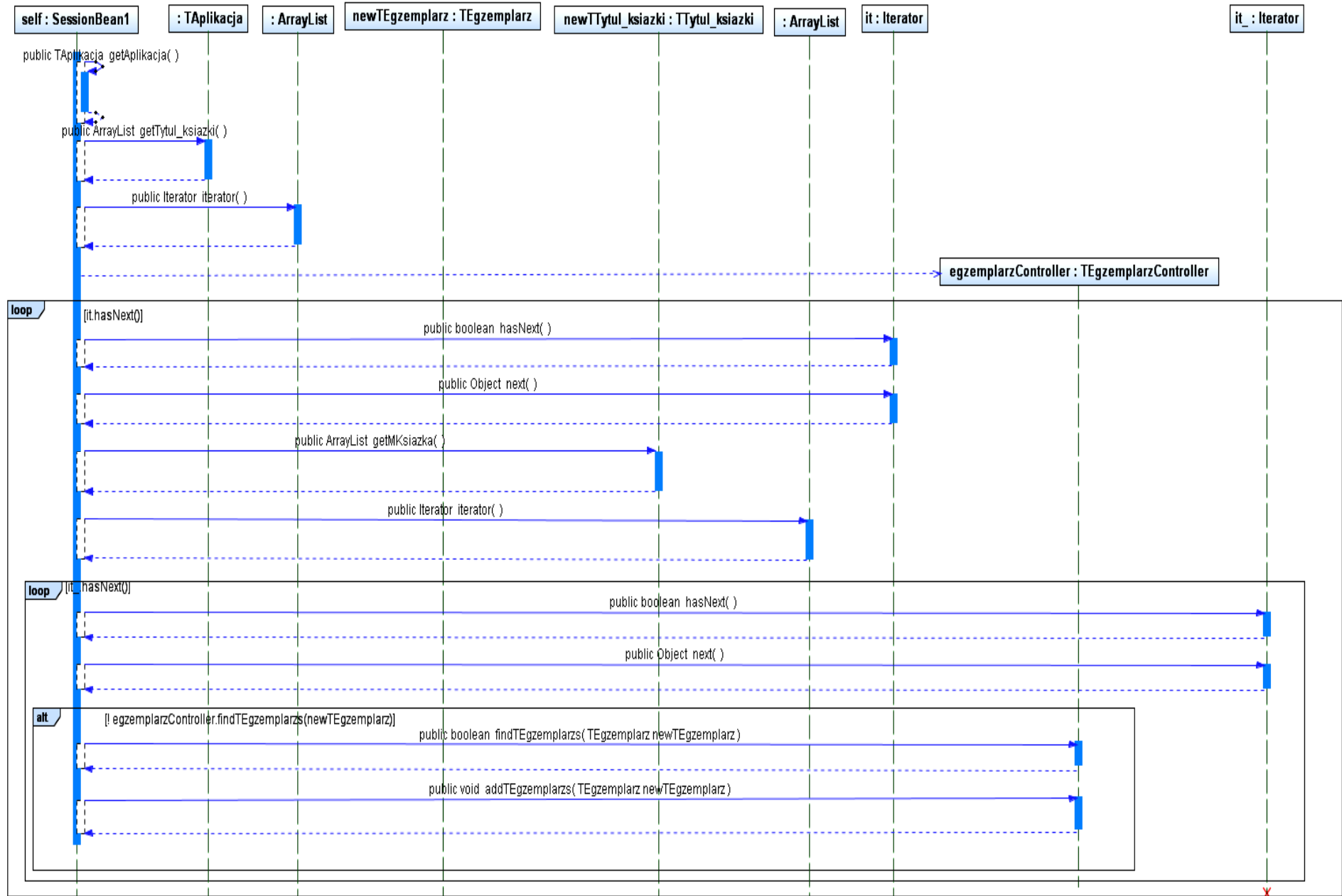
```
public boolean findTEgzemplarz(TEgzemplarz TEgzemplarz)
{
    EntityManager em = getEntityManager();
    try {
        return em.find(TEgzemplarz.class, TEgzemplarz.getId())!=null; }
    finally {
        em.close();
    }
}
```

Uzupełniono kod konstruktora klasy TEgzemplarz – wstawia klucz główny równy 0 do danej, która jeszcze nie została zapisana do bazy danych. Wartość 0 klucza głównego jest wartością nie należącą do dziedziny wartości kluczy głównych- wtedy można przekazać taki obiekt do metody find w klasie EntityManager

```
public TEgzemplarz() {
    id = new Long(-1); // id = null; dla wersji 2 w lab.7_8
}
```

Zapis danych do bazy książek metodą add obiektu TEgzemplarzController (20)

public void SessionBean1::zapisz_ksiazki_do_bazy()



Uruchomienie aplikacji (21)

http://localhost:8080/WebWypożyczalnia1/ - Windows Internet Explorer

http://localhost:8080/WebWypożyczalnia1/

Norton™ Monitorowanie fałszywych witryn jest włączone

Tytuły					
id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki			
id	numer	termin	tytuł_książki
No items found.			

Zapisz tytuły do bazy

Zapisz egzemplarze do baz

Podaj tytuł

Podaj autora

Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Podaj numer

Podaj termin

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1

Gotowe Internet | Tryb chroniony: włączony 100%

Wstawianie danych z aplikacji do tabeli **TEgzemplarz** w bazie danych (1) (22)

The screenshot shows a web browser window displaying a web application. The address bar shows the URL: `http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp`. The page contains two tables and several buttons.

Tytuły

id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki

id	numer	termin	tytuł_książki
52	1		Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
53	2	Sat Apr 19 00:00:00 CEST 2008	Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1

Buttons: **Zapisz tytuły do bazy**, **Zapisz egzemplarze do baz**, **Dodaj tytuł**, **Dodaj egzemplarz**

Form fields:

- Podaj tytuł:
- Podaj autora:
- Podaj ISBN:
- Podaj wydawnictwo:
- Podaj aktora:
- Podaj numer:
- Podaj termin:

Dropdown menus:

- Selected: Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
- Selected: Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Numer: 1
- Options: Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1, Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1

System status: Gotowe, Internet | Tryb chroniony: włączony, 100%

Wstawianie danych z aplikacji do tabeli **TEgzemplarz** w bazie danych (2) (23)

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp - Windows Internet Explorer

http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp

Monitorowanie fałszywych witryn jest włączone

Tytuły					
id	tytul	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki			
id	numer	termin	tytul_książki
52	1		Tytul: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
53	2	Sat Apr 19 00:00:00 CEST 2008	Tytul: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1

Zapisz tytuły do bazy

Zapisz egzemplarze do baz

Podaj tytuł

Podaj autora

Podaj ISBN

Podaj wydawnictwo

Podaj aktora

Podaj numer

Podaj termin

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1 Numer: 1 termin: Fri Apr 18 15:15:33 CEST 2008

Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1 Numer: 1 termin: Fri Apr 18 15:15:33 CEST 2008

Dodaj tytuł

Dodaj egzemplarz

Gotowe

Internet | Tryb chroniony: włączony

100%

Wstawianie danych z aplikacji do tabeli **TEgzemplarz** w bazie danych (3) (24)

The screenshot displays a web application interface in Internet Explorer. The browser address bar shows the URL: `http://localhost:8080/WebWypożyczalnia1/faces/Page1.jsp`. The page content includes two data tables and several buttons.

Tytuły

id	tytuł	autor	ISBN	wydawnictwo	aktor
2	1	1	1	1	
3	1	1	1	1	1

Książki

id	numer	termin	tytuł_książki
52	1		Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
53	2	Sat Apr 19 00:00:00 CEST 2008	Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1
102	1	Fri Apr 18 00:00:00 CEST 2008	Tytuł: 1 Autor: 1 ISBN: 1 Wydawnictwo: 1 Aktor: 1

Buttons:

Form fields for adding new data:

Podaj tytuł:

Podaj autora: Podaj numer:

Podaj ISBN: Podaj wydawnictwo: Podaj termin:

Podaj aktora:

Buttons:

Bottom status bar: Gotowe | Internet | Tryb chroniony: włączony | 100%

Widok bazy danych w zakładce **Services** – wykonanie operacji **ViewData** dostępnej z wyskakującego menu po naciśnięciu tabeli **TEgzemplarz** prawym klawiszem myszy

The screenshot shows an IDE interface with a database schema on the left and a query result on the right. The schema tree on the left shows a database with tables, sequences, indexes, and foreign keys. The **TEgzemplarz** table is highlighted. The query result on the right shows the following data:

ID	DTYPE	NUMER	MTYTUL_K...	TERMIN
52	TEgzemplarz	1	2	NULL
53	TEgzemplarz...	2	2	2008-04-19
102	TEgzemplarz...	1	3	2008-04-18
202	TEgzemplarz...	1	152	2008-04-18
252	TEgzemplarz...	3	3	2008-04-21
302	TEgzemplarz...	3	2	2008-04-19
352	TEgzemplarz	2	152	NULL
402	TEgzemplarz	4	152	NULL
452	TEgzemplarz	5	152	NULL
453	TEgzemplarz	4	2	NULL
502	TEgzemplarz	2	3	NULL
552	TEgzemplarz	4	3	NULL
602	TEgzemplarz	5	3	NULL
652	TEgzemplarz	6	3	NULL
653	TEgzemplarz	8	3	NULL
702	TEgzemplarz	9	3	NULL
752	TEgzemplarz	10	2	NULL
802	TEgzemplarz	12	2	NULL
902	TEgzemplarz	11	3	NULL
952	TEgzemplarz	7	3	NULL
1002	TEgzemplarz	13	3	NULL
1052	TEgzemplarz	14	3	NULL
1102	TEgzemplarz	15	3	NULL
1152	TEgzemplarz	16	3	NULL
1202	TEgzemplarz	17	3	NULL
1252	TEgzemplarz	18	3	NULL
1302	TEgzemplarz	19	3	NULL
1303	TEgzemplarz	20	3	NULL
1353	TEgzemplarz	10	3	NULL