Zaawansowane aplikacje internetowe - laboratorium

Java Persistence.

Do wykonania ćwiczeń potrzebne jest zintegrowane środowisko programistyczne NetBeans IDE 5.5 wraz z serwerem Sun Java System Application Server Platform Edition 9 (do pobrania z http://www.netbeans.org/downloads/index.html jako Java EE 5 Tools Bundle) oraz środowisko J2SE w wersji 1.5 Update 1 (lub wyższej) wymagane do instalacji NetBeans. Instalując Java EE 5 Tools Bundle należy zainstalować wszystkie składniki wraz z zawartym w pakiecie serwerem aplikacji (wybór opcji "Install the bundled Java EE SDK" na jednym z ekranów instalatora).

Ćwiczenie 1

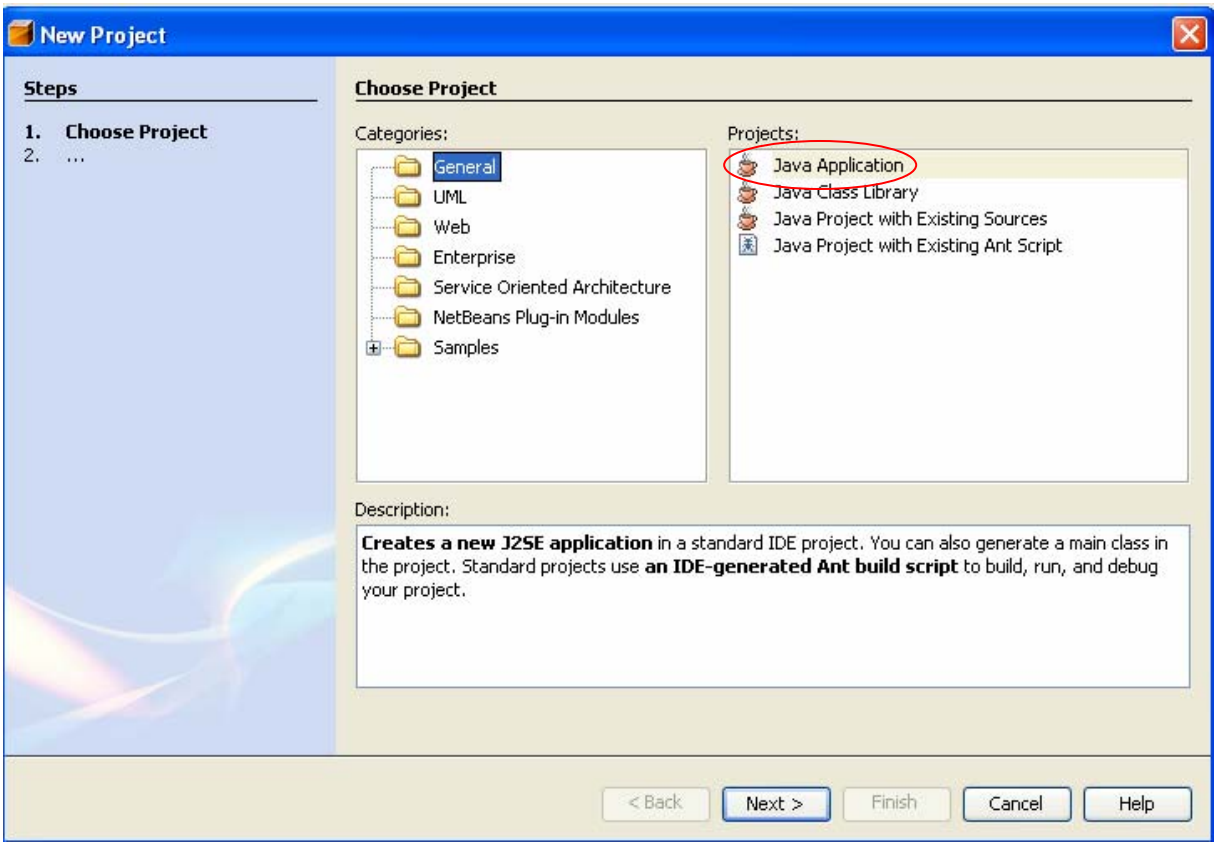
Celem ćwiczenia jest przygotowanie prostej aplikacji Java SE realizującej odczyt i zapis danych z/do bazy danych poprzez Java Persistence API.

Kroki ćwiczenia:

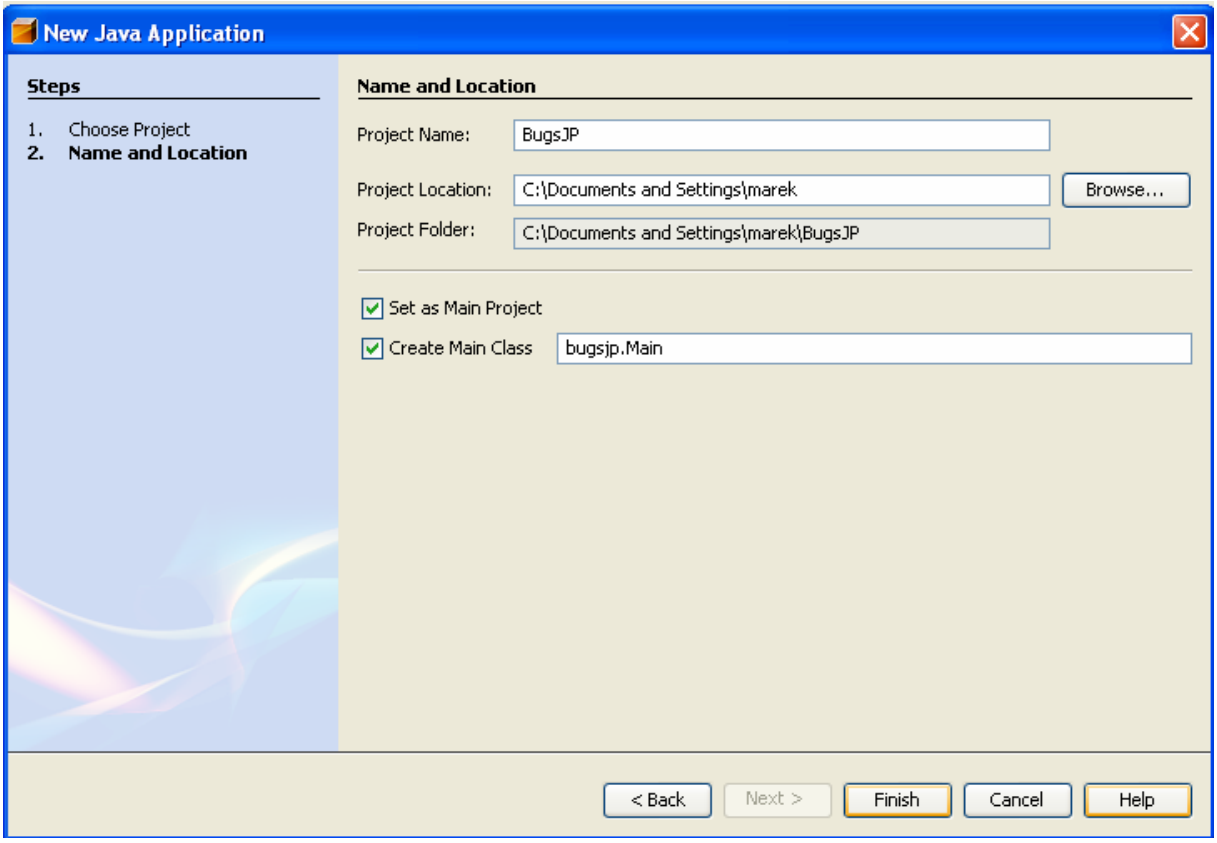
1. Utworzenie nowego projektu

a) Uruchom narzędzie NetBeans IDE 5.5

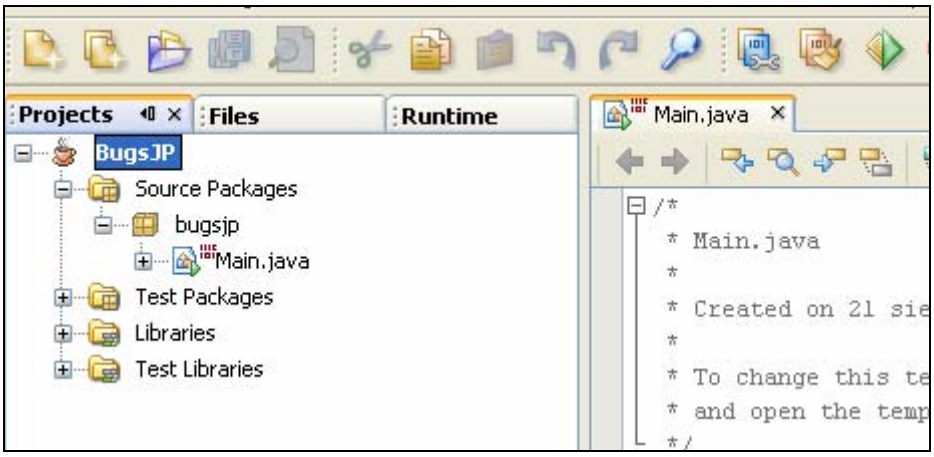
b) Z menu głównego wybierz File→New Project. Wybierz kategorię General i typ projektu Java Application. Kliknij przycisk Next >.



c) Podaj nazwę projektu „BugsJP”. W polu Project Location powinien być wskazany katalog, w którym masz prawo zapisu. Pola wyboru Set as Main Project i Create Main Class powinny być zaznaczone. Kliknij przycisk Finish.

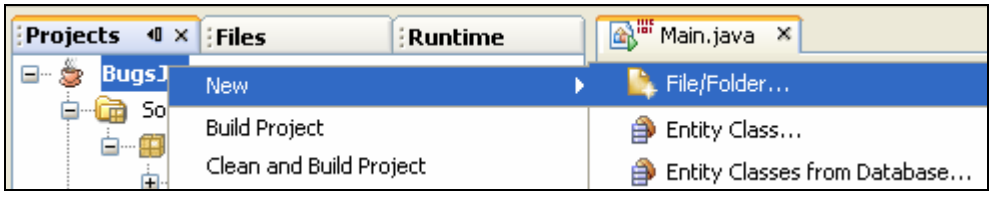


d) Efektem działania kreatora powinien być projekt zawierający klasę Java z metodą main().

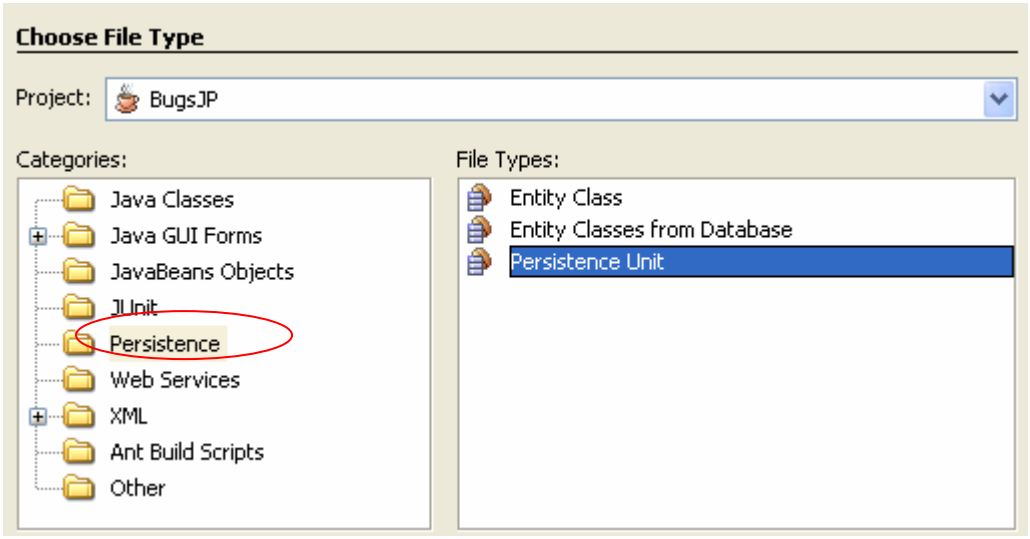


2. Utworzenie jednostki trwałości, w ramach której obiekty aplikacji będą zachowywane w bazie danych

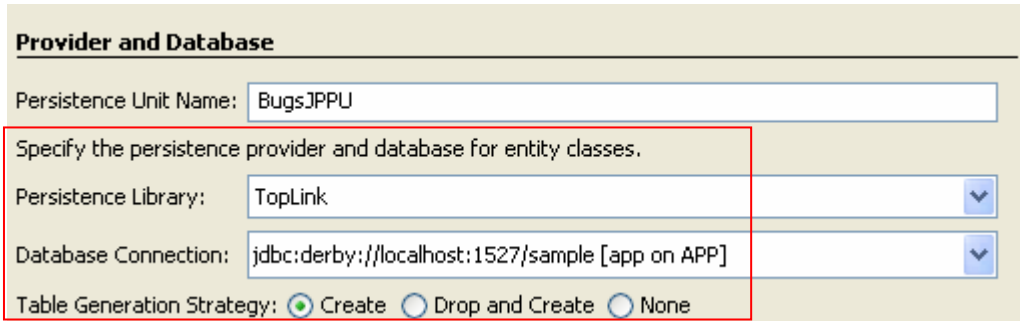
a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.



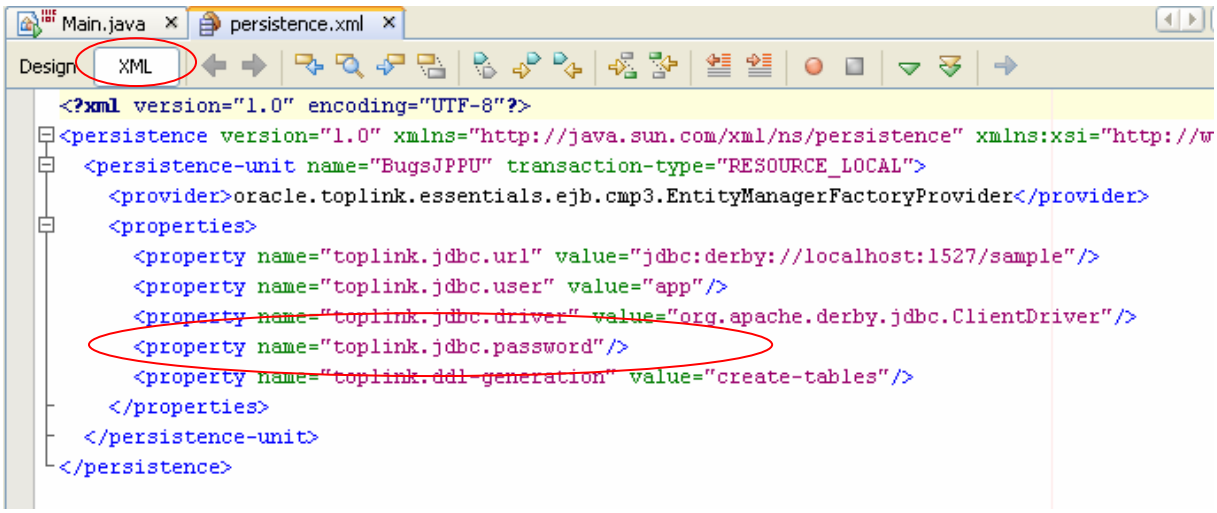
b) Następnie wybierz kategorię Persistence i typ pliku Persistence Unit.



c) W kolejnym oknie pozostaw wszystkie opcje domyślne. Zwróć uwagę na wybór biblioteki do obsługi trwałości (Persistence Library): TopLink i łańcuch połączenia JDBC (Database Connection), wskazujący bazę danych „wbudowaną" w środowisko NetBeans (Derby). Jako strategię tworzenia tabel w bazie danych (Table Generation Strategy) pozostaw Create, czyli tworzenie w momencie instalacji aplikacji jeśli nie istnieją. Kliknij przycisk Finish.



d) Obejrzyj zawartość wygenerowanego przez kreator pliku `persistence.xml` w trybie XML. Zwróć uwagę na elementy <provider> i <property> zawierające wartości odpowiadające opcjom wybranym w oknie kreatora. Sprawdź czy zarówno właściwość toplink.jdbc.user i toplink.jdbc.password mają wartość „app" i jeśli nie to popraw zawartość pliku.

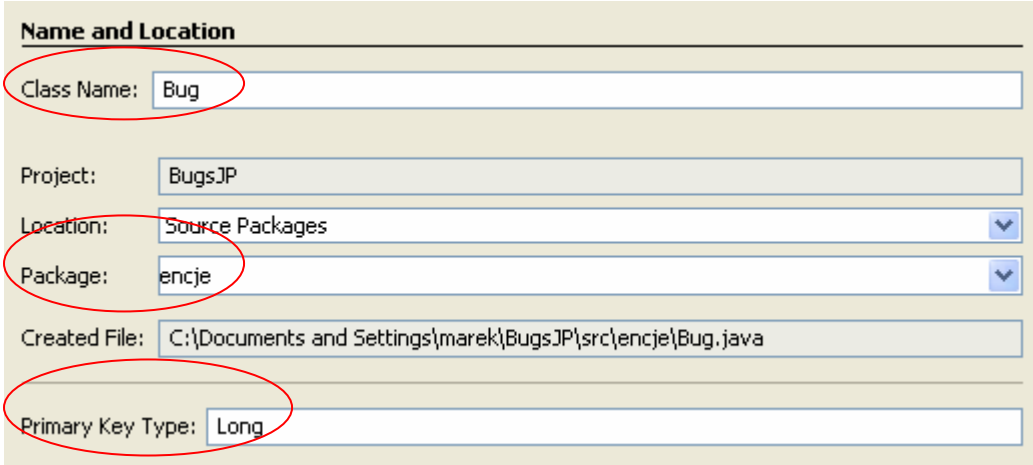


3. Utworzenie klasy encji Bug. Z każdym albumem będzie związany jeden wykonawca. Każdy wykonawca będzie posiadał kolekcję albumów.

a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.

b) Następnie wybierz kategorię Persistence i typ pliku Entity Class. Kliknij przycisk Next >.

c) Jako nazwę klasy podaj Bug, a jako nazwę pakietu encje. Pozostaw Long jako typ klucza głównego (Primary Key Type). Kliknij przycisk Finish.



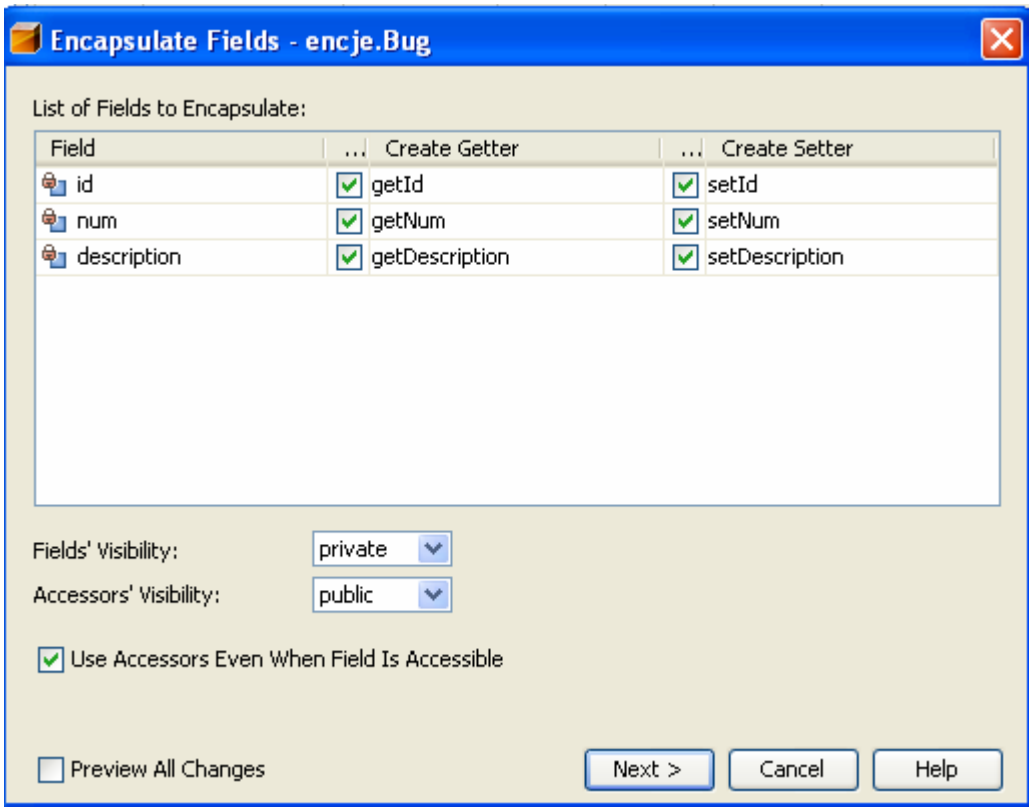
d) Ponownie obejrzyj zawartość pliku `persistence.xml` w trybie XML. Zwróć uwagę na element <class> dodany przez kreator tworzenia klasy encji. Atrybut ten oznacza, że utworzona klasa encji została wskazana jako zarządzana klasa trwała dla jednostki trwałości BugsJPPU.

e) Przejdź do edycji utworzonego pliku `Bug.java`. Pod adnotacją @Entity dodaj wiersz z adnotacją @Table(name="BUGS"), aby obiekty klasy były składowane w tabeli o nazwie BUGS (domyślnie nazwa tabeli byłaby taka jak nazwa klasy – w liczbie pojedynczej). Jeśli wprowadzona adnotacja zostanie podkreślona jako błąd, będąc kursorem w wierszu z adnotacją naciśnij kombinację klawiszy Alt-Enter i wybierz zaproponowaną opcję Add import for javax.persistence.Table.

f) Dodaj w klasie Bug (poniżej pola id) dwa prywatne pola num typu String i description typu String.

```
@Entity
@Table(name="BUGS")
public class Bug implements Serializable {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String num;
    private String description;
...
```

g) W oknie edycji klasy Bug prawym klawiszem myszy wywołaj menu kontekstowe i wybierz opcję Refactor→Encapsulate fields w celu utworzenia w klasie metod set/get dla pól. Upewnij się, że pola wyboru dla wszystkich metod set/get są zaznaczone. Odznacz pole Preview all changes, aby zmiany nie wymagały potwierdzenia i kliknij przycisk Next >.



h) Obejrzyj w kodzie klasy wygenerowane metody.

i) Zdefiniuj w klasie encji Bug nazwane zapytanie do wyszukiwania błędów zawierających w opisie podane słowo kluczowe poprzez umieszczenie bezpośrednio po wierszu z adnotacją @Table wiersza z poniższą adnotacją @NamedQuery:

```
    @NamedQuery(name = "findByKeyword", query = "SELECT b FROM Bug b WHERE
b.description LIKE :keyword")
```

4. Dodanie w klasie Main kodu zapisującego obiekty Bug do bazy danych i odczytującego obiekty Bug z bazy danych.

a) Przejdź do edycji pliku `Main.java`. W oknie edycji klasy Main prawym klawiszem myszy wywołaj menu kontekstowe i wybierz opcję Persistence→ Use Entity Manager. Operacja ta spowoduje dodanie w klasie Main kodu tworzącego obiekt

EntityManagerFactory i metody persist() ilustrującej sposób tworzenia obiektu EntityManager i realizacji transakcji.

b) Dodaj do kodu klasy Main dyrektywy import importujące klasy z pakietu encje i klasę biblioteczną java.util.List:

```
import encje.*;
import java.util.List;
```

c) Zastąp kod metodę persist() poniższą metodą addBug(), tworzącą obiekt Bug na podstawie podanego numeru i opisu, a następnie zapisującą go do bazy danych.

```
public static void addBug(String pNum, String pDesc) {
    EntityManager em = emf.createEntityManager();
    em.getTransaction().begin();
    try {
      Bug b = new Bug();
      b.setNum(pNum);
      b.setDescription(pDesc);
      em.persist(b);
      em.getTransaction().commit();
    } catch (Exception e) {
        e.printStackTrace();
        em.getTransaction().rollback();
    } finally {
        em.close();
    }
}
```

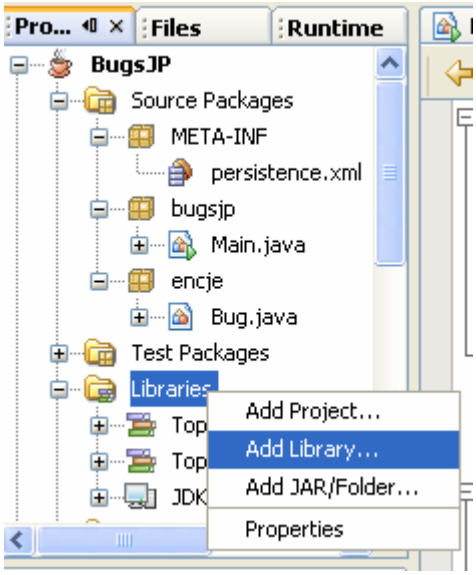
d) Dodaj w klasie Main metodę findBugs(), wyszukującą w bazie danych błędy, których opisy zawierają podane słowo kluczowe, poprzez nazwane zapytanie zdefiniowane w klasie encji.

```
public static List<Bug> findBugs(String pKeyword) {
    EntityManager em = emf.createEntityManager();
    List<Bug> wyn = null;
    try {
      wyn = em.createNamedQuery("findByKeyword")
              .setParameter("keyword", pKeyword).getResultList();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        em.close();
    }
    return wyn;
}
```

e) Dodaj w metodzie main() klasy Main kod zapisujący do bazy danych kilka obiektów Bug metodą addBug(), a następnie wyszukujący obiekty Bug metodą findBugs() i wyświetlający wyniki wyszukiwania na konsoli, np.:

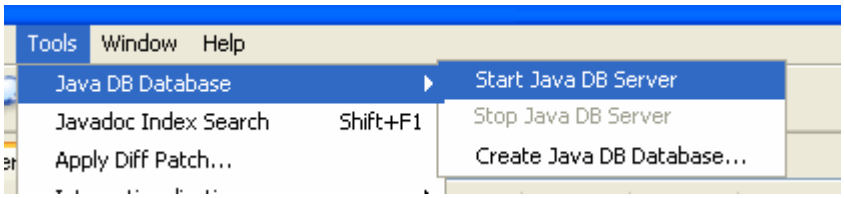
```
public static void main(String[] args) {
    addBug("b001", "Database server process does not start");
    addBug("b002", "Database open() does not work");
    addBug("b003", "Execution slow when on battery");
    for (Bug b : findBugs("%base%")) {
      System.out.println("Num:" + b.getNum() +
                         " Desc: " + b.getDescription());
    }
}
```

f) Dodaj do projektu bibliotekę wymaganą do współpracy z wbudowaną w NetBeans bazą danych. W tym celu wywołaj dla węzła Libraries w drzewie projektu opcję Add Library, a następnie z listy dostępnych bibliotek wybierz Java DB Driver i kliknij przycisk Add Library.



j) Zapisz wszystkie zmiany (File→Save All lub ikona w pasku narzędzi).

g) Uruchom wbudowany serwer bazy danych wybierając z menu głównego opcję Tools→Java DB Database → Start Java DB Server.



h) Uruchom aplikację wybierając opcję Run Project z menu kontekstowego dla projektu. Wyświetlone na konsoli dane odczytane z bazy danych mogą być poprzedzone komunikatami informacyjnymi i ostrzeżeniami zgłaszanymi przez Toplink.

Ćwiczenie 2

Celem ćwiczenia jest:

1) przygotowanie modelu obiektowego dla aplikacji obejmującego dwie powiązane ze sobą encje i odwzorowanie go w schemat relacyjnej bazy danych;

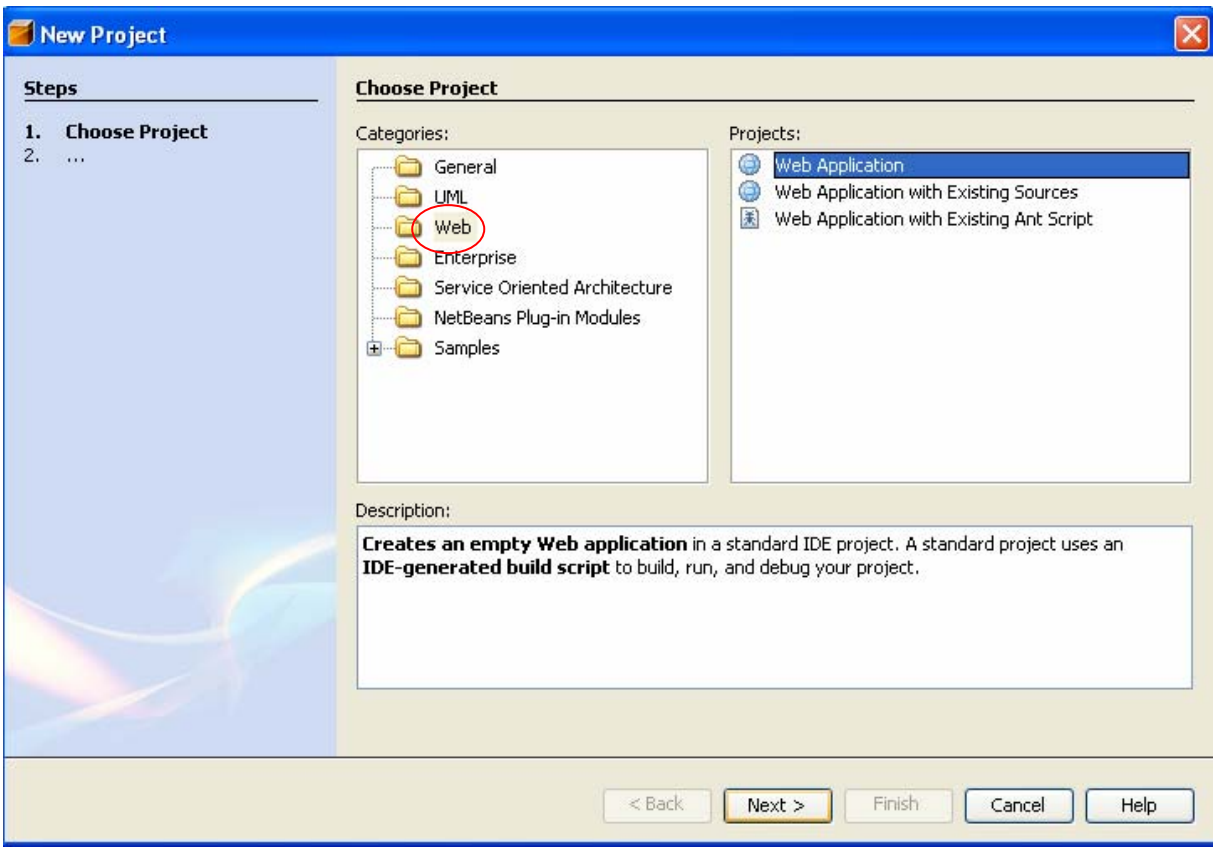
2) wygenerowanie za pomocą kreatora aplikacji Java EE opartej o JSF pracującej na utworzonym modelu i obsługującej trwałość poprzez Java Persistence API.

Kroki ćwiczenia:

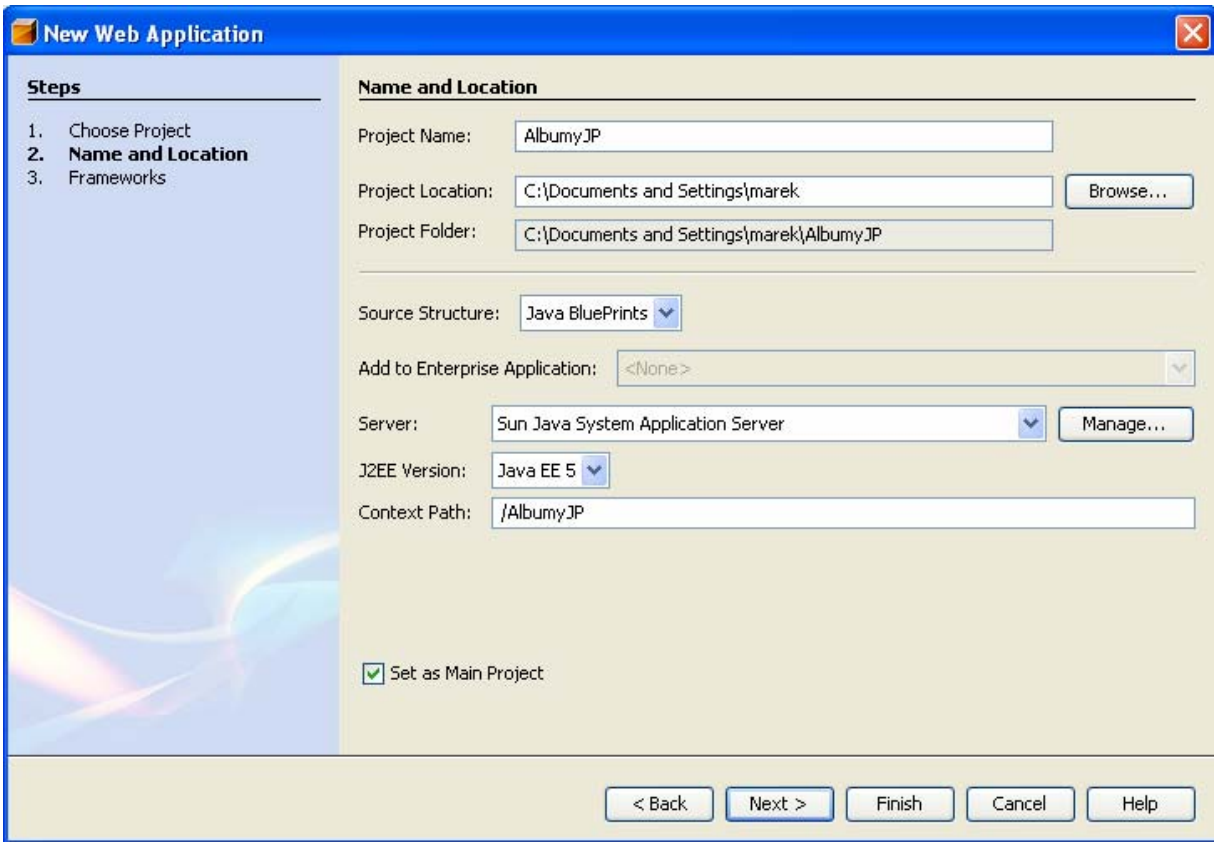
1. Utworzenie nowego projektu

a) Uruchom narzędzie NetBeans IDE 5.5

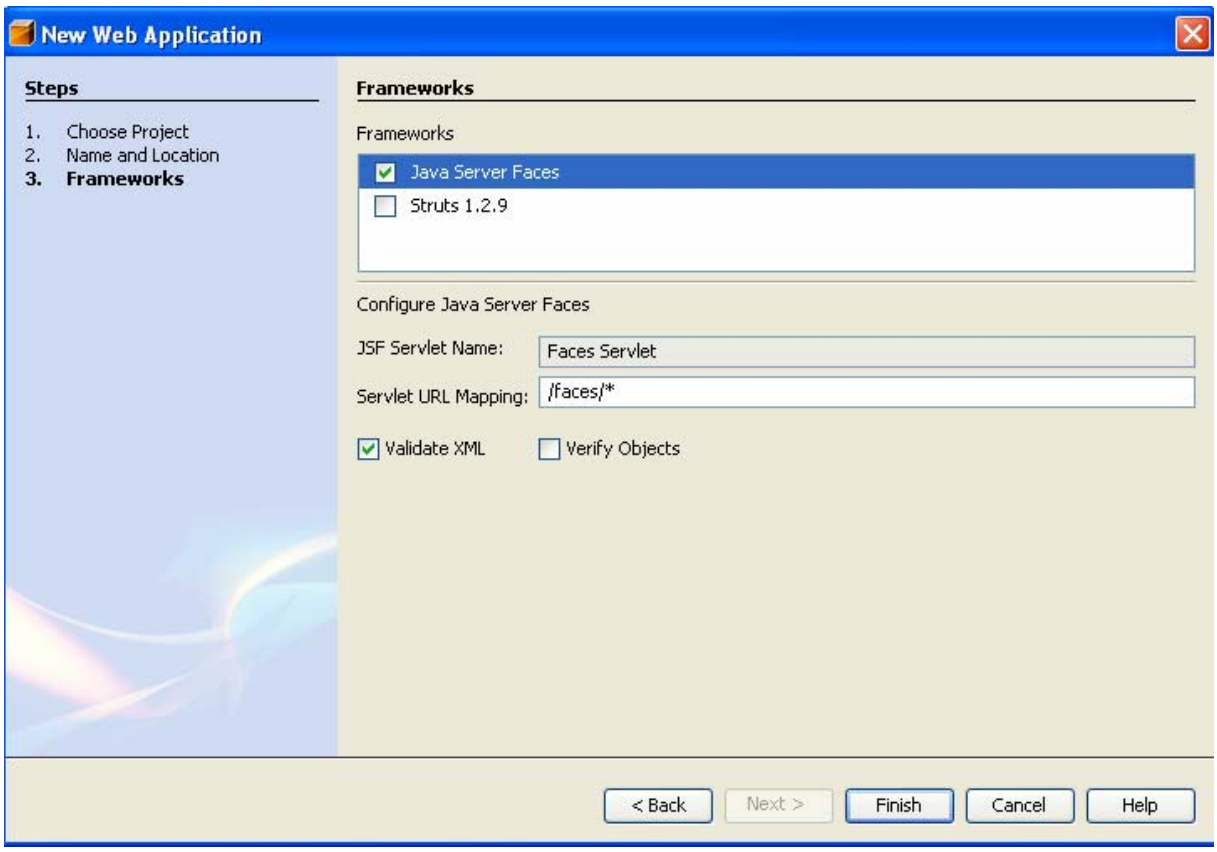
b) Z menu głównego wybierz File→New Project. Wybierz kategorię Web i typ projektu Web Application. Kliknij przycisk Next >.


New Project
Steps
1. Choose Project
2. ...
Choose Project
Categories:
General
UML
Web
Enterprise
Service Oriented Architecture
NetBeans Plug-in Modules
Samples
Projects:
Web Application
Web Application with Existing Sources
Web Application with Existing Ant Script
Description:
Creates an empty Web application in a standard IDE project. A standard project uses an IDE-generated build script to build, run, and debug your project.
< Back
Next >
Finish
Cancel
Help

c) Podaj nazwę projektu, „AlbumyJP”. Zwróć uwagę, że wraz zmianą nazwy projektu zmienia się Context Path czyli katalog wirtualny na serwerze WWW, który będzie prowadził do aplikacji. W polu Project Location powinien być wskazany katalog, w którym masz prawo zapisu. Jako Server powinien być wybrany Sun Java System Application Server a jako J2EE Version – Java EE 5. Kliknij przycisk Next >.

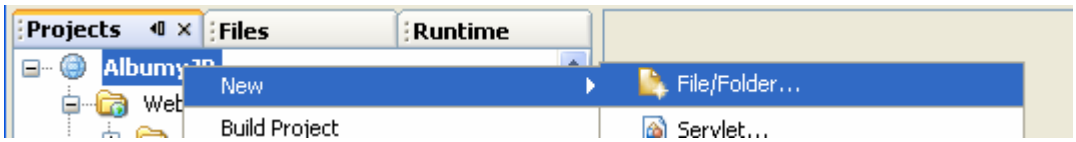


d) Zaznacz Java Server Faces w panelu Frameworks. Krok ten jest niezbędny, gdyż w ostatnim etapie ćwiczenia do prezentacji danych z bazy danych będzie wykorzystana aplikacja WWW oparta o JSF. Kliknij przycisk Finish.

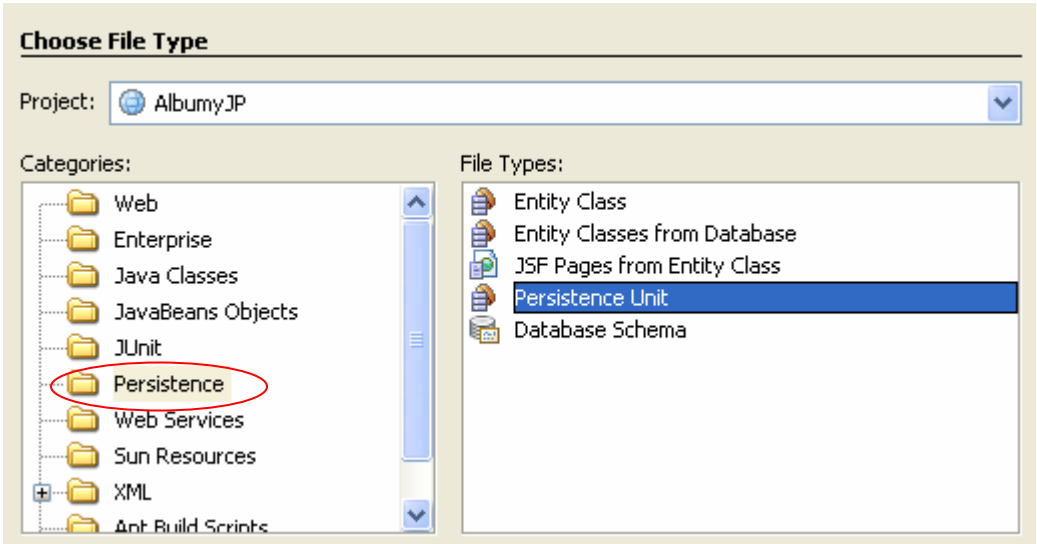


2. Utworzenie jednostki trwałości, w ramach której obiekty aplikacji będą zachowywane w bazie danych

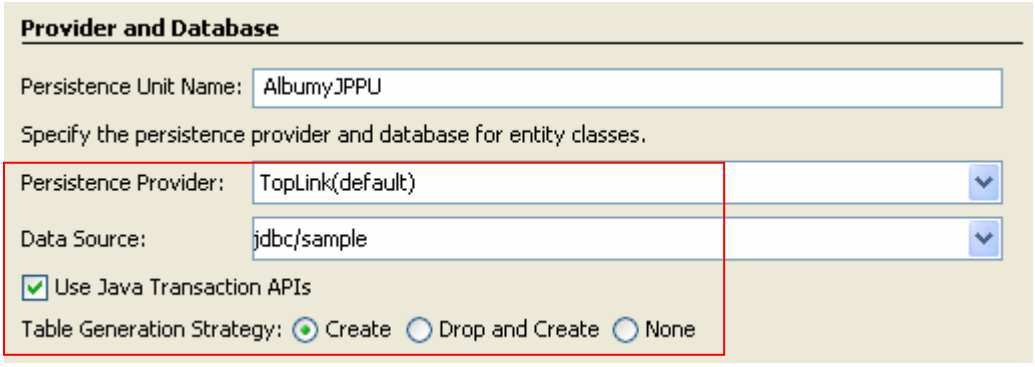
a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.



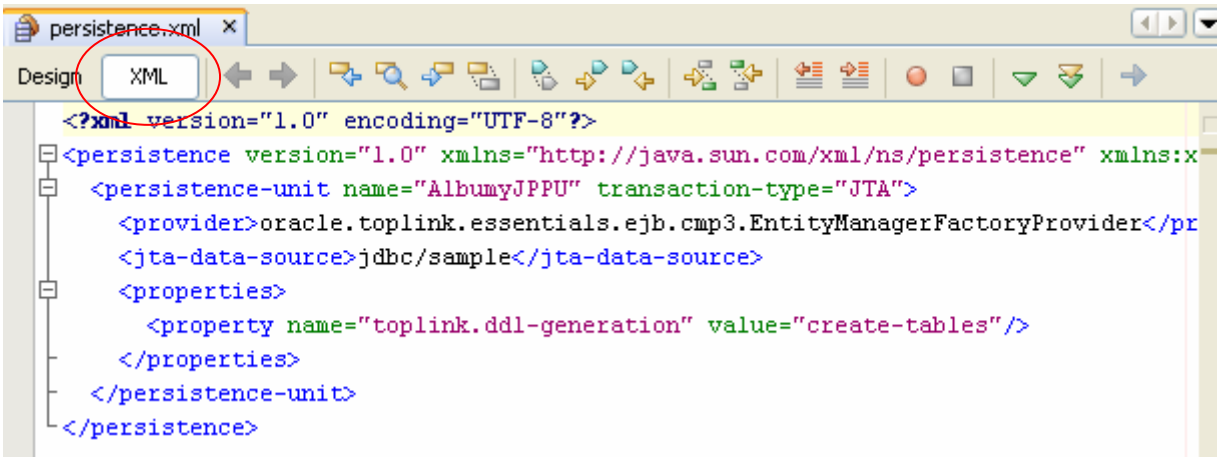
b) Następnie wybierz kategorię Persistence i typ pliku Persistence Unit.



c) W kolejnym oknie pozostaw wszystkie opcje domyślne. Zwróć uwagę na wybór dostawcy usług trwałości (Persistence Provider): TopLink (alternatywą jest m.in. Hibernate) i źródło danych jdbc/sample, reprezentujące bazę danych „wbudowaną" w środowisko NetBeans. Upewnij się, że pole wyboru Use Java Transaction API jest zaznaczone. Jako strategię tworzenia tabel w bazie danych (Table Generation Strategy) pozostaw Create, czyli tworzenie w momencie instalacji aplikacji jeśli nie istnieją. Kliknij przycisk Finish.



d) Obejrzyj zawartość wygenerowanego przez kreator pliku `persistence.xml` w trybie XML. Zwróć uwagę na elementy <provider> i <jta-data-source> zawierające wartości odpowiadające opcjom wybranym z list rozwijanych w oknie kreatora.

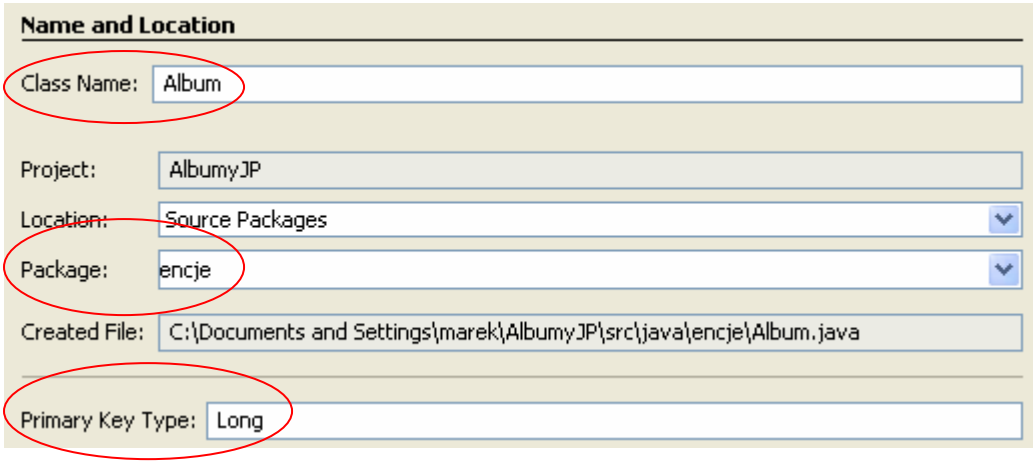


3. Utworzenie klas encji Album i Wykonawca. Z każdym albumem będzie związany jeden wykonawca. Każdy wykonawca będzie posiadał kolekcję albumów.

a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.

b) Następnie wybierz kategorię Persistence i typ pliku Entity Class. Kliknij przycisk Next >.

c) Jako nazwę klasy podaj „Album", a jako nazwę pakietu „encje". Pozostaw Long jako typ klucza głównego (Primary Key Type). Kliknij przycisk Finish.



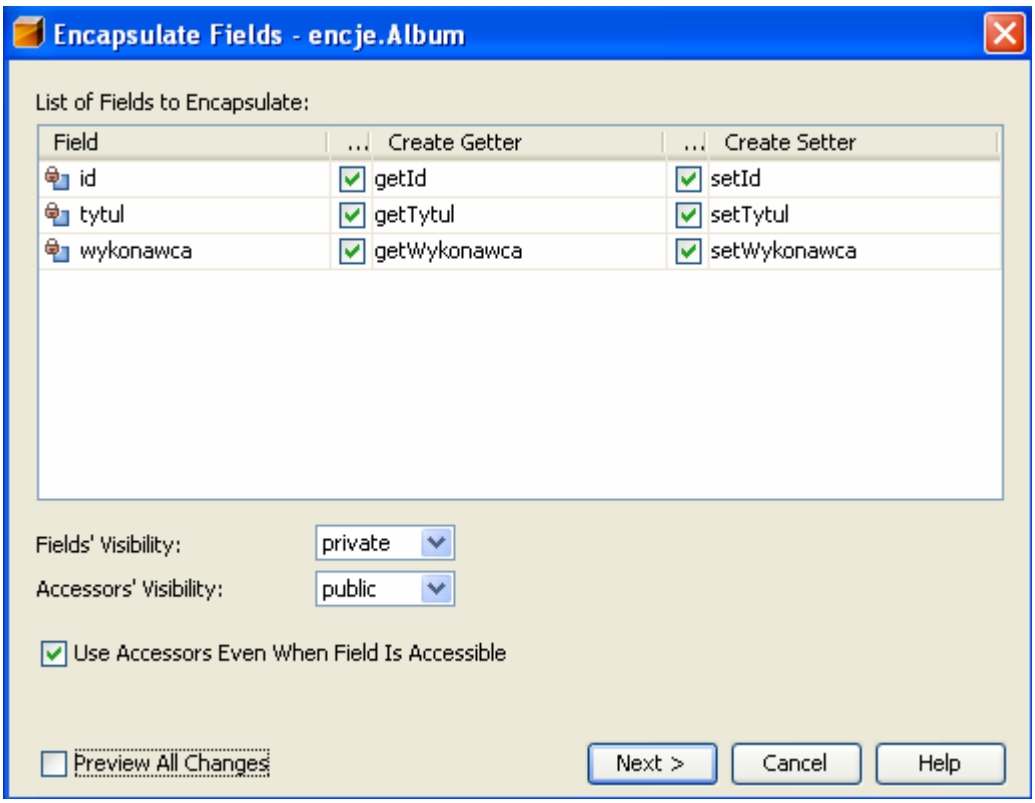
d) Przejdź do edycji utworzonego pliku `Album.java`. Pod adnotacją @Entity dodaj wiersz z adnotacją @Table(name="ALBUMY"), aby obiekty klasy były składowane w tabeli o nazwie ALBUMY (domyślnie nazwa tabeli byłaby taka jak nazwa klasy – w liczbie pojedynczej). Jeśli wprowadzona adnotacja zostanie podkreślona jako błąd, będąc kursorem w wierszu z adnotacją naciśnij kombinację klawiszy Alt-Enter i wybierz zaproponowaną opcję Add import for javax.persistence.Table.

e) Dodaj w klasie Album (poniżej pola id) dwa prywatne pola tytu" typu String i wykonawca typu Wykonawca. Nie przejmuj się tym, że pole wykonawca zostało oznaczone jako błąd. Błąd ten wynika z faktu użycia jako typu pola klasy Wykonawca, która jeszcze nie została utworzona.

```
@Entity
@Table(name="ALBUMY")
public class Album implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String tytul;
    private Wykonawca wykonawca;
...
```

f) W oknie edycji klasy Album prawym klawiszem myszy wywołaj menu kontekstowe i wybierz opcję Refactor→Encapsulate fields w celu utworzenia w klasie metod set/get dla pól. Upewnij się, że pola wyboru dla wszystkich metod set/get są zaznaczone. Odznacz pole Preview all changes, aby zmiany nie wymagały potwierdzenia i kliknij przycisk Next >.



g) Obejrzyj w kodzie klasy wygenerowane metody.

h) W celu utworzenia drugiej klasy, ponownie kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.

i) Następnie wybierz kategorię Persistence i typ pliku Entity Class. Kliknij przycisk Next >.

j) Jako nazwę klasy podaj Wykonawca, a jako nazwę pakietu encje. Pozostaw Long jako typ klucza głównego (Primary Key Type). Kliknij przycisk Finish.

k) Przejdź do edycji utworzonego pliku `Wykonawca.java`. Pod adnotacją @Entity dodaj wiersz z adnotacją @Table(name="WYKONAWCY"). Zaimportuj klasę javax.persistence.Table podobnie jak wcześniej dla klasy Album.

l) Dodaj w klasie Wykonawca (poniżej pola id) dwa prywatne pola nazwa typu String i albumy typu Collection<Album> (nie zapomnij o zaimportowaniu java.util.Collection).

```
@Entity
@Table(name="WYKONAWCY")
public class Wykonawca implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String nazwa;
    private Collection<Album> albumy;

...
```

m) W oknie edycji klasy Wykonawca prawym klawiszem myszy wywołaj menu kontekstowe i wybierz opcję Refactor→Encapsulate fields w celu utworzenia w klasie

metod set/get dla pól. Upewnij się, że pola wyboru dla wszystkich metod set/get są zaznaczone. Odznacz pole Preview all changes, aby zmiany nie wymagały potwierdzenia i kliknij przycisk Next >.

4. Zdefiniowanie związku 1:N między klasami encji Wykonawca i Album

   a) Oznacz w klasie Wykonawca pole albumy adnotacją @OneToMany(mappedBy="wykonawca"). Zaimportuj klasę adnotacji jak wcześniej dla adnotacji @Table.

```
@OneToMany(mappedBy="wykonawca")
private Collection<Album> albumy;
```

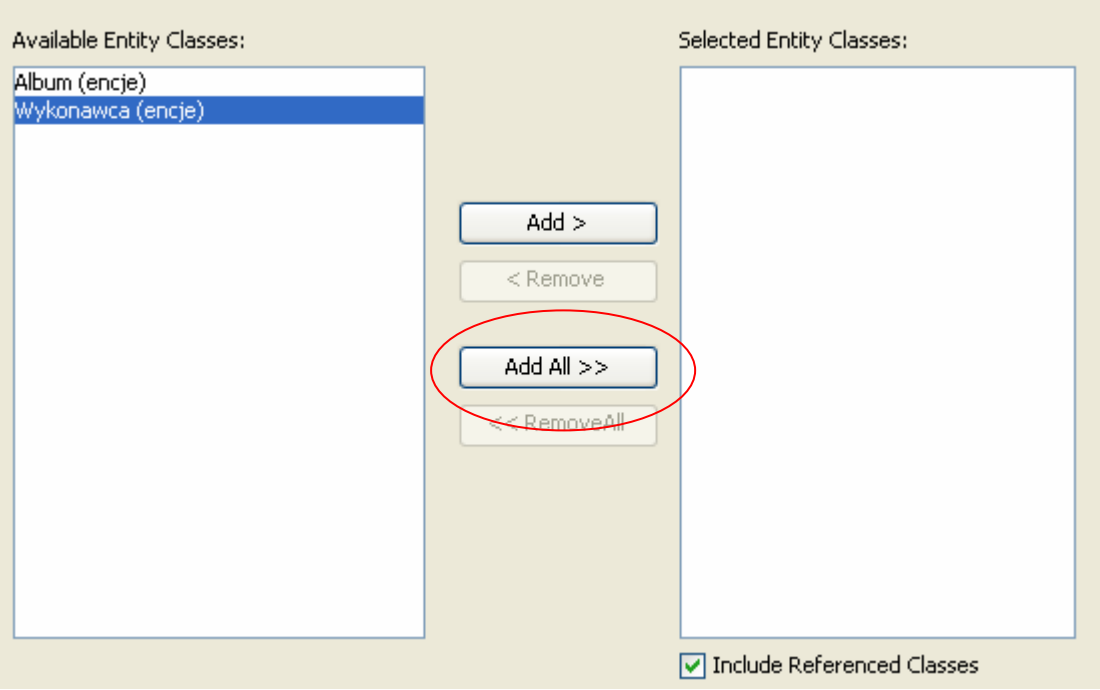
   b) Oznacz w klasie Album pole wykonawca adnotacją @ManyToOne. Zaimportuj klasę adnotacji.

```
@ManyToOne
private Wykonawca wykonawca;
```

5. Ostatnia część ćwiczenia to wygenerowanie za pomocą kreatora aplikacji JSF do

   a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→JSF Pages from Entity Class. Kliknij przycisk Next >.

   b) Wybierz wszystkie encje jako źródło dla aplikacji JSF klikając przycisk Add All.

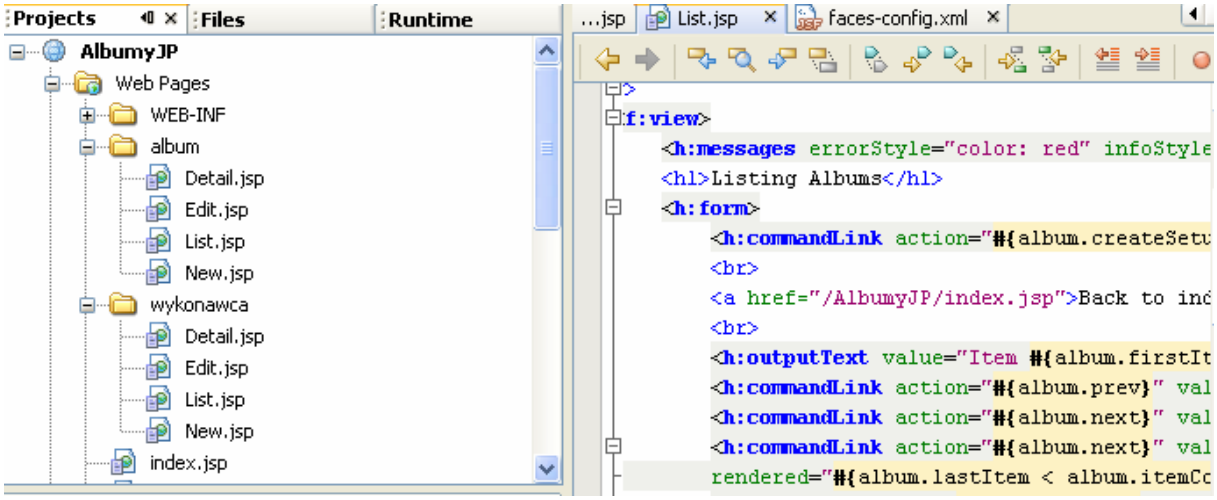


   c) Kliknij przycisk Next >. W kolejnym oknie pozostaw wszystkie ustawienia domyślne i kliknij przycisk Finish.

   d) Wynikiem działania kreatora jest zestaw czterech stron JSF dla każdej encji i towarzyszące im klasy pomocnicze – po dwie dla każdej encji. Rozwiń wszystkie gałęzie drzewa projektu w oknie Projects, aby obejrzeć wygenerowane pliki.

   e) Dla każdej encji wygenerowane zostały cztery strony JSF (`Detail.jsp`, `Edit.jsp`, `List.jsp`, `New.jsp`), umożliwiające przeglądanie, edycję, dodawanie i usuwanie wykonawców i albumów. W celu zapoznania się z efektem działania

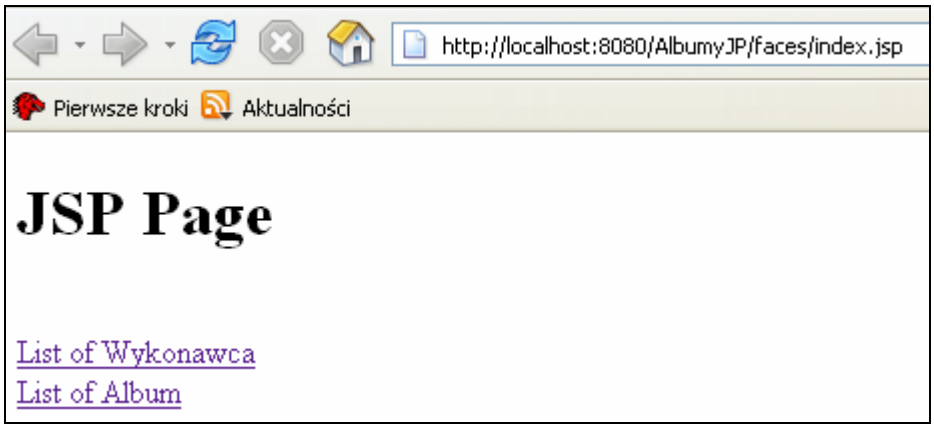
kreatora otwórz plik `album/List.jsp`. Jest to strona JSF, której komponenty interfejsu odwołują się do zarządzanego komponentu JavaBean o nazwie album. Następnie otwórz plik konfiguracyjny aplikacji JSF `faces-config.xml`. Znajdź deklarację zarządzanego komponentu album. Jest on obiektem wygenerowanej przez kreator klasy AlbumController obsługującej komunikację z bazą danych dla encji Album. Podejrzyj źródło klasy AlbumController. Znajdź w klasie adnotacje wstrzykujące obiekty UserTransaction i EntityManagerFactory. Obejrzyj wywołania metod Java Persistence API w metodach klasy.



f) Usuń z drzewa projektu wygenerowaną wraz z projektem przykładową stronę JSF `welcomeJSF.jsp`. W tym celu prawym klawiszem myszy wywołaj menu kontekstowe dla pliku w drzewie projektu, a następnie wybierz opcje Delete.

g) Przejdź do edycji pliku `index.jsp` i usuń link do strony `welcomeJSF.jsp`. Zapisz wszystkie dokonane zmiany (File→Save All lub ikona w pasku narzędzi).

h) Uruchom aplikację wybierając opcję Run File z menu kontekstowego dla pliku `index.jsp`.



i) Dodaj dwóch wykonawców, a następnie po dwa albumy dla każdego z nich. Przykładowo, w celu dodania wykonawcy wybierz link List of wykonawca, następnie New Wykonawca, wprowadź nazwę i wybierz link Create.

Uwaga: Wygenerowana aplikacja zawiera teksty w języku angielskim. Oczywiście można zmodyfikować kod poszczególnych stron tłumacząc teksty nagłówków i opisy linków na język polski.

Ćwiczenie 3

Celem ćwiczenia jest wygenerowanie biblioteki klas encji dla istniejącego schematu relacyjnej bazy danych. Taka biblioteka może następnie być wykorzystywana i współdzielona przez aplikacje Java SE i Java EE działające na bazie danych.

Kroki ćwiczenia:

1. Utworzenie nowego projektu

   a) Uruchom narzędzie NetBeans IDE 5.5

   b) Z menu głównego wybierz File→New Project. Wybierz kategorię General i typ projektu Java Class Library. Kliknij przycisk Next >.

   c) Podaj nazwę projektu „OrdersLibrary”. W polu Project Location powinien być wskazany katalog, w którym masz prawo zapisu. Kliknij przycisk Finish.

2. Utworzenie jednostki trwałości, w ramach której obiekty aplikacji będą zachowywane w bazie danych

   a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.

   b) Następnie wybierz kategorię Persistence i typ pliku Persistence Unit.

   c) W kolejnym oknie pozostaw wszystkie opcje domyślne. Zwróć uwagę na wybór biblioteki do obsługi trwałości (Persistence Library): TopLink i łańcuch połączenia JDBC (Database Connection), wskazujący bazę danych „wbudowaną” w środowisko NetBeans (Derby). Jako strategię tworzenia tabel w bazie danych (Table Generation Strategy) pozostaw Create, czyli tworzenie w momencie instalacji aplikacji jeśli nie istnieją. Kliknij przycisk Finish.

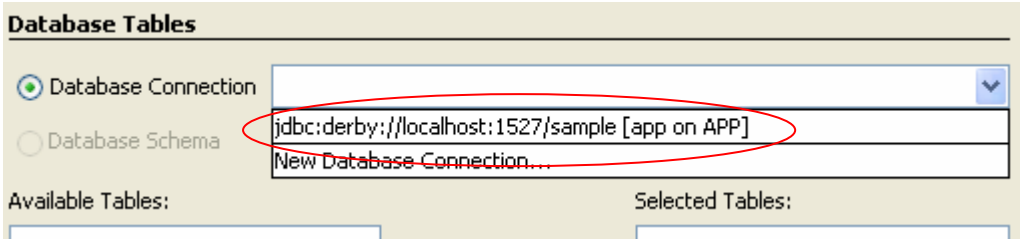
   d) Obejrzyj zawartość wygenerowanego przez kreator pliku `persistence.xml` w trybie XML. Zwróć uwagę na elementy <provider> i <property> zawierające wartości odpowiadające opcjom wybranym w oknie kreatora. Sprawdź czy zarówno właściwość toplink.jdbc.user i toplink.jdbc.password mają wartość „app” i jeśli nie to popraw zawartość pliku.

3. Utworzenie za pomocą kreatora zbioru encji na podstawie istniejących tabel w bazie danych.

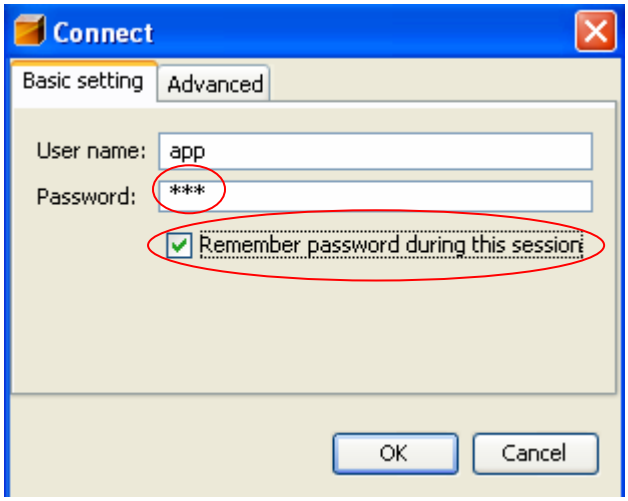
a) Kliknij prawym przyciskiem myszy na ikonie projektu w drzewie projektów i z menu kontekstowego wybierz New→File/Folder. Kliknij przycisk Next >.

b) Następnie wybierz kategorię Persistence i typ pliku Entity Classes from Database. Kliknij przycisk Next >.

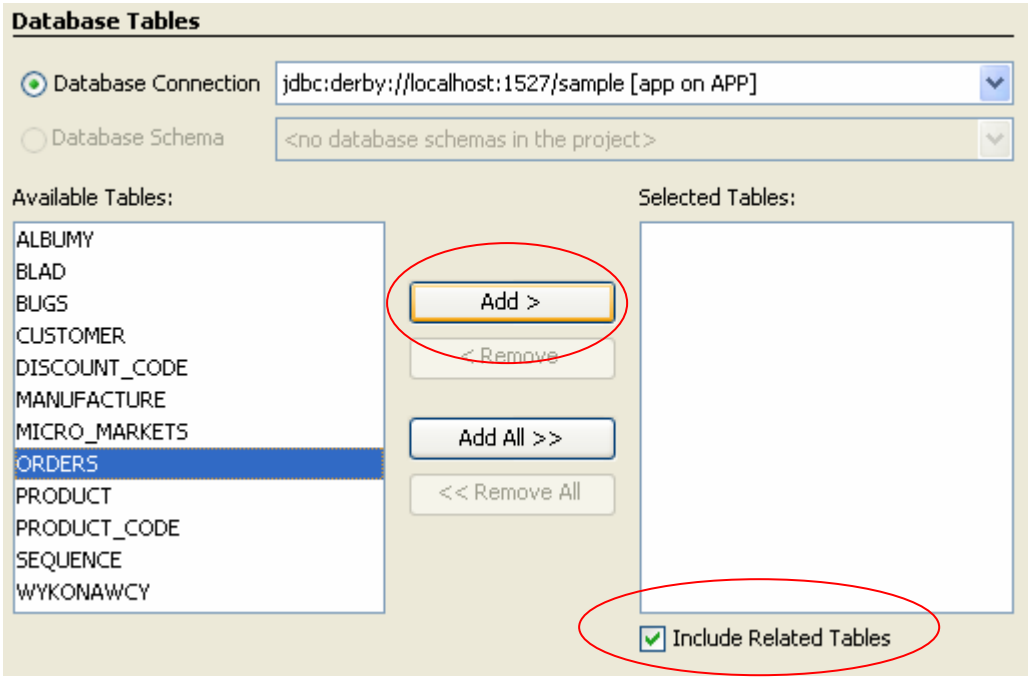
c) Z listy zdefiniowanych w środowisku NetBeans połączeń z bazą danych wybierz połączenie z wbudowanym serwerem Derby



d) Jeśli pojawi się okienko z prośbą o podanie hasła wprowadź hasło „app”, zaznacz pole wyboru Remember password during this session i kliknij przycisk OK.



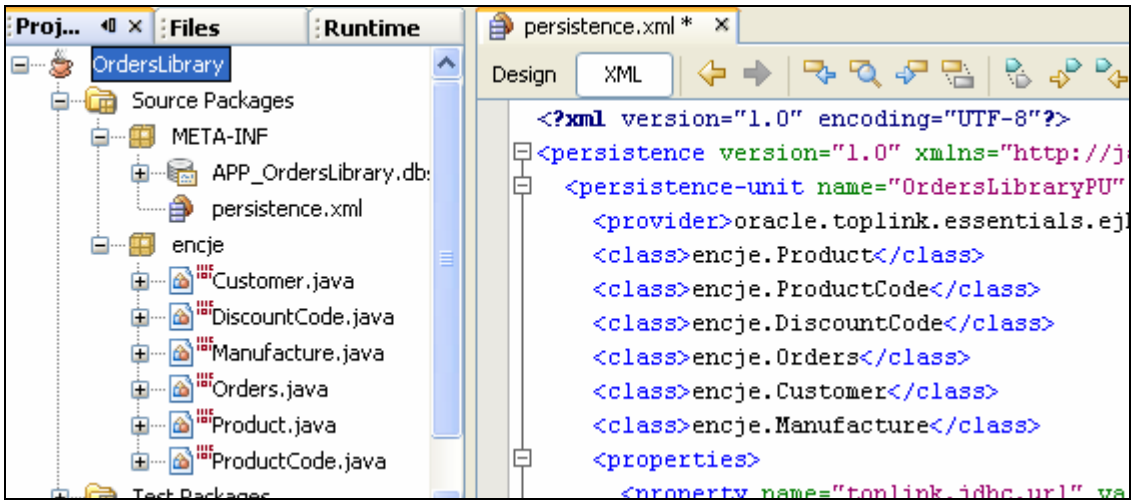
e) Z listy tabel do wyboru, przy zaznaczonym polu wyboru Include related tables, zaznacz myszą tabelę ORDERS i kliknij przycisk Add >.



f) Zwróć uwagę, że dzięki opcji Include Related Tables automatycznie zostały wybrane wszystkie tabele tworzące sieć połączeń zawierającą wskazaną tabelę ORDERS. Kliknij przycisk Next >.

g) W kolejnym oknie kreatora jako nazwę pakietu wprowadź „encje". Pozostaw zaproponowane nazwy klas encji dla poszczególnych tabel i pozostałe opcje. Kliknij przycisk Finish.

h) Obejrzyj wynik działania kreatora. Rozwiń drzewo projektu i obejrzyj wygenerowane klasy encji. Obejrzyj zawartość pliku `persistence.xml` w trybie XML. Zwróć uwagę na elementy <class> dodane przez kreator, wskazujące wygenerowane klasy jako zarządzane klasy trwałe dla jednostki trwałości OrdersLibraryPU.



i) Zbuduj projekt, wybierając z menu kontekstowego węzła projektu opcję Build Project. Wynikiem powinien być plik JAR ze skompilowaną biblioteką.