|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Architektura zorientowana na usługi** | | | | | | | | | | | |
| 4 | Temat: | ***WebServices*** | Zadania: | | | | | | | | Data: |
| Autor: | **Sylwia Kaleta** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | **07 XI 2018** |
| Autor: | **Kamil Wanat** | M | E | E | M | - | - | - | - | **18:00-19:30** |

**Zadanie 1. Tworzenie usługi WebServices**

Celem zadania było zaimplementowanie WebService’u według specyfikacji podanej przez prowadzącego w instrukcji laboratoryjnej. WebService miał na celu obsługę prostej listy zawierającej klasę wraz z informacją o nauczycielu oraz uczniach. Lista została zaimplementowana z użyciem EJB (dokładnie Stateles Bean) co pozwoliło na przechowywanie wymaganych informacji przez cały okres trwania sesji. Metody wymagane w zadaniu zostały zaimplementowane w klasie WebService’u. Poniżej widoczna jest implementacja metody pobierającej konkretną klasę z listy wszystkich klas:

@WebMethod(operationName = "getConcreteClass")

public String getConcreteClass(@WebParam(name = "number") String number,

@WebParam(name = "letter") String letter)

{

int classNumber = Integer.parseInt(number);

StudentClass studentClass = classesContainer.findClass(classNumber, letter);

if (studentClass==null)

return "Nie ma takiej klasy";

return studentClass.toString();

}

**Zadanie 2. Analiza WSDL**

W tym zadaniu należało wygenerować (lub własnoręcznie napisać plik WSDL). Środowisko NetBeans wspiera w tym programiste i pozwala na automatyczne wygenerowanie pliku WSDL za pomocą jednego kliknięcia. Niestety w wersji NetBeans 8.2 (uzywanej w trakcie wykonywania zadania” polecenie wykonywane w programie NetBeans jest błędne, co prowadzi do błędu podczas generowania pliku. Aby wygenerować plik WSDL należy przekopiować polecenie zwracane przez błąd NetBeans, następnie dołożyć spację pomiędzy polecenie wsgen a ścieżkę do pliku oraz uruchomić ręcznie polecenie z konsoli.

WSDL to język znaczników XML wykorzystywany do opisu technicznych parametrów WebService. Najważniejsze znaczniki to: <service> wraz ze znacznikami <port> definiują adresy punktów dostępowych dla usługi. Znaczniki <portType> służą do deklaracji funkcji biznesowych oferowanych przez usługę. Znaczniki <binding> określają metody kodowania parametrów wywołania i parametrów zwrotnych usługi. <Operation> oraz <Messager> definiują kolejno operację wykonywaną oraz komunikat przesyłany poprzez WebService

Poniżej znajduje się fragment pliku WSDL z naszego projektu:

*<message name="addStudentToClass">*

*<part name="parameters" element="tns:addStudentToClass"/>*

*</message>*

*<portType name="Lab4WebService">*

*<operation name="addStudentToClass">*

*<input wsam:Action="http://lab4.com/Lab4WebService/addStudentToClassRequest" message="tns:addStudentToClass"/>*

*<output wsam:Action="http://lab4.com/Lab4WebService/addStudentToClassResponse" message="tns:addStudentToClassResponse"/>*

*</operation>*

*<binding name="Lab4WebServicePortBinding" type="tns:Lab4WebService">*

*<soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>*

*<operation name="addStudentToClass">*

*<soap:operation soapAction=""/>*

*<input>*

*<soap:body use="literal"/>*

*</input>*

*<output>*

*<soap:body use="literal"/>*

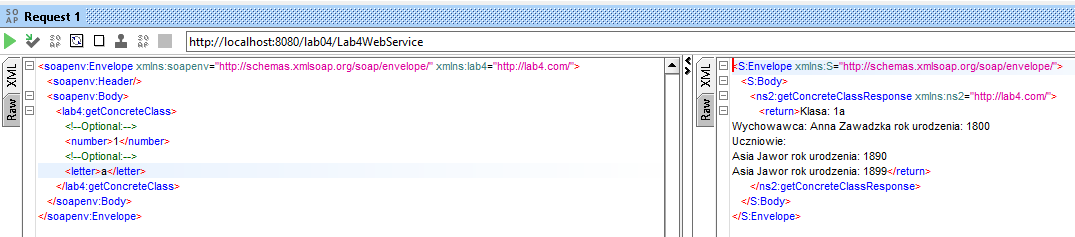
*</output>*

*</operation>*

W powyższym pliku fragment *<soap:binding style="document"/>* wskazuje na wykorzystanie http do przesyłania danych natomiast *<soap:body use="literal"/>* wskazuje na użyty typ danych. Operacje zdefiniowane w pliku WSDL zgodne są z funkcjami implementowanymi w implementacji WebService’u tak samo jak dane wejściowe oraz wyjściowe.

**Zadanie 3. Testowanie usługi**

Celem zadania było przetestowanie usługi WebService przy użyciu zewnętrznego narzędzia jakim jest SoapUI. Po utworzeniu nowego projektu SOAP w programie zostajemy poproszeni o podanie ścieżki do pliku WSDL. NA jego podstawie program przygotuje da użytkownika odpowiednie żądania. Wystarczy jedynie dodać interesujące nas dane a następnie wysłać żądanie. Odpowiedź WebService’u otrzymujemy w równoległym oknie, dzięki czemu możemy porównać wysłane żądanie oraz wiadomość zwrotną. Poniżej przedstawiono zrzut ekranu z programu ukazujący pobranie konkretnej klasy poprzez podanie jej numeru i litery:



**Zadanie 4. Klient WebService**

Klienta WebService zaimplementowaliśmy w języku Java z wykorzystaniem wbudowanych mechanizmów środowiska NetBeans. Dzięki temu implementacja klienta staje się dużo prostsza i sprowadza się jedynie do odpowiedniego wywołania metod wygenerowanych przez środowisko. Poniżej znajduje się kod funkcji usuwającej wybraną klasę, oraz wygenerowana funkcja obsługująca połączenie z WebServicem:

private static String removeCl() {

Scanner scann = new Scanner(System. in);

System.out.println("Podaj numer klasy:");

String number = scann.nextLine();

System.out.println("Podaj litere klasy:");

String letter = scann.nextLine();

return removeClass(number,letter);

}

private static String removeClass(java.lang.String number, java.lang.String letter) {

com.lab4.Lab4WebService\_Service service = new com.lab4.Lab4WebService\_Service();

com.lab4.Lab4WebService port = service.getLab4WebServicePort();

return port.removeClass(number, letter);

}

**Podsumowanie**

Celem ćwiczenia było zapoznanie się z technologią SOAP WebService’ow. Rozwiązanie zadań laboratoryjnych pozwoliło nam na lepsze zrozumienie działania WebService oraz rozpoznanie różnić pomiędzy REST a SOAP. Dzięki temu w przyszłości będziemy mogli podjąć świadomą decyzję której technologi należy użyć w danym projekcie. Dzięki wykorzystaniu środowiska NetBeans część zadań udało się nieco zautomatyzować, co ułatwiło dalszą pracę z projektem.