

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з дисципліни

«Архітектура і технології Web-служб»

для студентів спеціальності 123 «Комп'ютерна інженерія»

всіх форм навчання

Частина 4

Web-служби на основі технології JAX-WS

2019

Методичні вказівки до лабораторних робіт з дисципліни «Архітектури і технології Web-служб» для студентів спеціальності 123 «Комп'ютерна інженерія» всіх форм навчання. Частина 4. Web-служби на основі технології JAX-WS / Укл.: С.Д. Точилін, А.В. Тіменко. – Запоріжжя: НУ «Запорізька політехніка», 2019. – 10 с.

Укладачі:

С.Д. Точилін, доцент, к.ф-м.н.
А.В. Тіменко, асистент

Рецензент:

І.Я. Зеленьова, доцент, к.т.н.

Відповідальний за випуск:

С.Д. Точилін, доцент, к.ф-м.н.

Затверджено
на засіданні кафедри
«Комп'ютерних систем та мереж»
Протокол № 1 від 21.08.2019

Рекомендовано до видання:
НМК факультету КНТ
Протокол № 1 від 23.08.2019

ЗМІСТ

1 ТЕХНОЛОГІЯ JAX-WS	4
1.1 Загальні відомості.....	4
1.2 Переваги JAX-WS.....	4
1.3 Анотації	5
2 СТВОРЕННЯ РОЗПОДІЛЕНОГО ДОДАТКУ НА ОСНОВІ JAX-WS-ТЕХНОЛОГІЇ.....	5
2.1 Створення Web-сервіса	5
2.2 Клієнт Web-сервіса.....	7
3 ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ.....	9
4 КОНТРОЛЬНІ ПИТАННЯ	10
5 РЕКОМЕНДОВАНА ЛІТЕРАТУРА	10

Лабораторна робота №4 **«WEB-служба на основі технології JAX-WS»**

Мета роботи: ознайомитися з технологією JAX-WS. Навчитися створювати розподілені додатки на основі технології JAX-WS.

1 ТЕХНОЛОГІЯ JAX-WS

1.1 Загальні відомості

JAX-WS (Java API for XML Web Services) - важливий компонент платформи Java EE.

Специфікація JAX-WS - наступний крок, у порівнянні зі специфікацією JAX-RPC, на шляху реалізації Java Web-сервісів. Технологія JAX-WS спрощує розробку Web-сервісів та їх клієнтів, взаємодіючих з використання SOAP-протоколу, приховуючи при цьому від розроблювача всі деталі обробки SOAP-повідомлень.

JAX-WS є заміною технології JAX-RPC, надаючи більш документо-орієнтовану модель повідомлень та спрощуючи розробку Web-сервісів за рахунок використання анотацій, що вперше з'явилися в Java SE 5.

Технологія JAX-WS без додаткових налаштувань забезпечує підтримку протоколів SOAP 1.1, SOAP 1.2 та XML/HTTP. Технологія JAX-WS інтегрована з технологією Java Architecture for XML Binding (JAXB), що забезпечує перетворення між XML-документами та Java-класами.

Крім того, JAX-WS дозволяє розробляти асинхронні Web-сервіси.

1.2 Переваги JAX-WS

1. Технологія JAX-WS підтримує транспортні протоколи, відмінні від HTTP.
2. Пряма інтеграція з JAXB 2.0.
3. JAX-WS дозволяє створювати Web-сервіси, орієнтовані як на повідомлення, так і на модель RPC.
4. Підтримка MTOM. Message Transmission Optimization Mechanism(MTOM) - це метод, який дозволяє більш ефективно передавати бінарні дані в SOAP-повідомленнях, використовуючи XOP (XMLbinary Optimized Packaging).

5. Можливість вибору між двома шляхами розробки: знизу-нагору (програміст розробляє endpoint-класи сам) та зверху-донизу (Java класи генеруються по WSDL).

1.3 Анотації

JAX-WS використовує анотації, які описані в JSR 1.81, для відображення класів на WSDL. Серед них:

1. **@WebService** - указує на те, що Java клас (або інтерфейс) є Web-сервісом.
2. **@WebMethod** - маркує метод як операцію Web-сервісу.
3. **@WebParam** - настраює зв'язок параметрів методу із частинами повідомлень та XML-XML-елементами.
4. **@WebResult** - настраює зв'язок значення, що повертається, із частинами повідомлень XML-елементами.
5. **@Oneway** - указує, що у даного методу **@WebMethod** є тільки вхідні параметри та немає вихідних параметрів.
6. **@SOAPBinding** - дозволяє настроїти те, як буде описуватися Web-сервіс із використанням протоколу SOAP.

2 СТВОРЕННЯ РОЗПОДІЛЕНОГО ДОДАТКУ НА ОСНОВІ JAX-WS-ТЕХНОЛОГІЇ

2.1 Створення Web-сервіса

Для створення Web-сервісу треба написати Java клас з використанням анотації **@WebService** та визначити методи, які будуть доступні клієнтам за допомогою анотації **@WebMethod**.

Створення **Web**-сервісу складається із трьох етапів:

1. Створення інтерфейсу (опис методів, які будуть доступні в Web-сервісі для клієнтів).
2. Реалізація інтерфейсу (клас, що реалізує інтерфейс і містить код методів, які виконують **Web**-послуги).
3. Публікація Web-сервісу.

Приклад створення Web-сервісу.

Створимо 3 java класи.

А). Інтерфейс **AddOperationInterface.java**.

В цьому файлі оголошується Web-сервіс за допомогою анотації `@WebService`, та для неї визначається унікальний ідентифікатор `http://com.zntu.jax-ws`. Крім того, оголошується Web-метод, який буде доступний клієнтам.

```
import javax.jws.WebMethod;
import javax.jws.WebService;

@WebService(targetNamespace= "http://com.zntu.jax-ws")
public interface AddOperationInterface{
    @WebMethod
        float add(float a, float b);
}
```

Б). Реалізація інтерфейсу - **AddOperationImpl.java**.

В **AddOperationImpl.java** вказується той же самий унікальний ідентифікатор `targetNamespace`, як і у файлі **AddOperationInterface.java**, крім того, визначається параметр `endpointInterface` та реалізуються інтерфейс сервіса.

```
import javax.jws.WebService;
import javax.jws.WebMethod;

@WebService(endpointInterface = "AddOperationInterface",
targetNamespace = "http://com.zntu.jax-ws")
public class AddOperationImpl implements AddOperationInterface
{
    public float add(float a, float b) {
        return a+b;
}
```

В). Клас для публікації Web-сервісу: **AddServicePublisher.java**.

Клас **AddServicePublisher.java**, визначає адресу Web-сервіса (`http://localhost:8080/ws/add_service`), та його реалізацію, яку ми розміщаємо по цій адресі.

```
import javax.xml.ws.Endpoint;

public class AddServicePublisher{
    public static void main(String[] args) {
        Endpoint.publish("http://localhost:8080/ws/add_service",
            new AddOperationImpl());}
    }
}
```

Компіляцію усіх файлів виконуємо за допомогою команди:
javac *.java.

У результаті компіляції маємо 3 нових файли формату *.class.

Запуск Web-сервісу здійснюється в окремій консолі командою:
java AddServicePublisher.

Якщо все було виконано правильно, то за адресою `http://localhost:8080/ws/add_service?wsdl` можна переглянути WSDL-опис Web-сервісу (дивись рисунок 2.1).

2.2 Клієнт Web-сервіса

Клієнт підключається до Web-сервісу на підставі wsdl-файлу, та екземпляра класу QName, який визначає компонент в wsdl-файлі, що описує Web-сервіс.

```
import java.net.URL;
import javax.xml.namespace.QName;
import javax.xml.ws.Service;

public class AddServiceClient{
    public static void main(String[] args) throws Exception {
        URL url = new URL("http://localhost:8080/ws/add_service?wsdl");
        QName qname = new QName("http://com.zntu.jax-ws",
            "AddOperationImplService");
        Service service = Service.create(url, qname);
        AddOperationInterface addService =
            service.getPort(AddOperationInterface.class);
        float a; float b;
        try { a = Float.parseFloat(args[0]); b = Float.parseFloat(args[1]);
        } catch (Exception e) { // No arguments. Setting default values.
```

```

    a = 8.0f; b = 2.0f;}
    float result = addService.add(a,b);
    System.out.println(" Received from service: " + result);
}

```

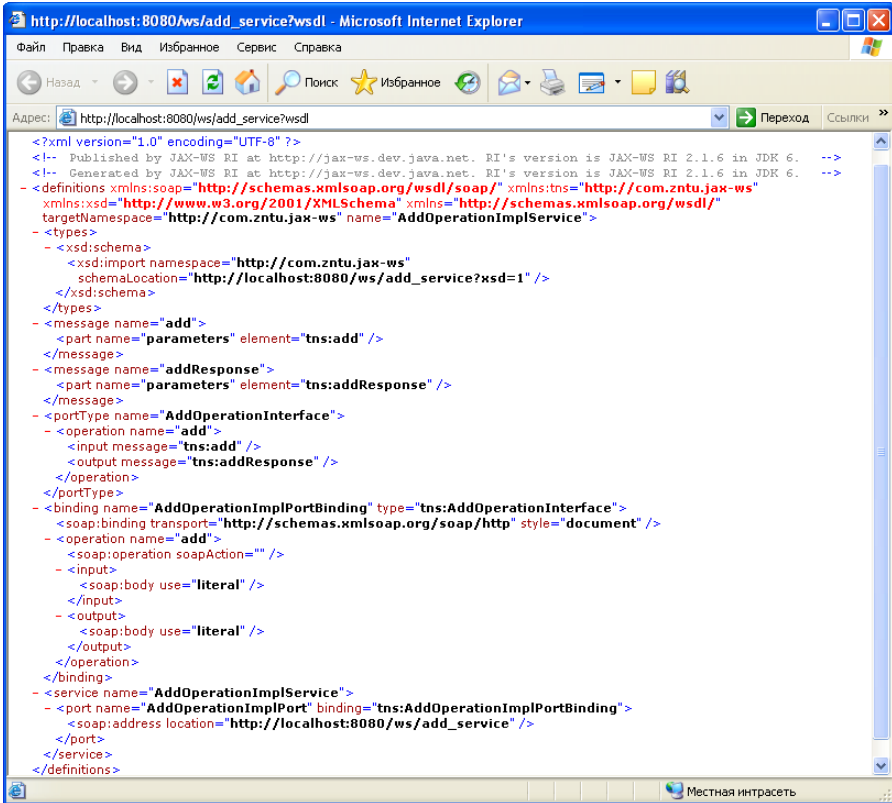


Рисунок 2.1 - WSDL-опис Web-сервісу

Відкомпілюємо файл: **javac AddServiceClient.java**
 Запустимо клієнта Web-сервіса: **java AddServiceClient 5 15**
 У результаті на консоль буде виведене повідомлення:
Received from service: 20.0

3 ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ

Завдання 1.

Створіть Web-сервіс, який описаний у п. 2.1.

Перегляньте його WSDL-опис за допомогою браузера.

Завдання 2.

Створіть клієнта для Web-сервіса, який описаний у п. 2.2.

Проведіть з його допомогою розподілені обчислення.

Завдання 3.

Створіть розподілений Java-додаток Web-сервіс якого при запиті клієнта робить обчислення у відповідності з математичною формулою Вашого варіанта. Клієнт передає серверу чисельні значення змінних, що визначені у формулі.

№ варіанта	Математична формула
1	$x + y^2 + z / j + \sqrt{y}$
2	$10x + 1 / y + 10 / z + \sqrt[3]{m}$
3	$[5a^{nx} / (b+c)] - (\cos x)^{3^{0.5}}$
4	$a + bx^3 + yz$
5	$\sqrt{a+b+c} + n^5$
6	$a + b^{0.5} + \ln x^3 + e^{2y}$
7	$ \sin x \cos \frac{x}{2} + \sqrt{y+z^2} + j$
8	$\sqrt[5]{x} (\operatorname{arctg} z + \cos^2 y) + k$
9	$\sqrt[4]{y + \sqrt{x-y} + z / k}$
10	$e^x \cos(y \sin z) + j$
11	$\sqrt{\ln(x - 2y \cos z + j^2)}$
12	$\sqrt{a_0 + a_1 x + a_2 \sqrt[3]{ \sin x }}$
13	$\frac{\operatorname{tg} x}{y} + \sqrt[3]{\sin z} + k$

14	$[x^{0.5}/y]+z/(\sin mx)^{0.5}$
15	$\frac{xyz}{(x+y)+(z+m)}$
16	$\ln x + \ln y + \sqrt{z} + \sqrt[3]{m}$
17	$x^y+z/(\cos n)^{0.2}$
18	$e^{-\frac{ah}{kT}} + 1$
19	$\sqrt{x^2 + 2yz + d^2}$
20	$x^{-1}+y^{-1}+z^{-1}+d^{-1}$

4 КОНТРОЛЬНІ ПИТАННЯ

1. У чому відмінність технологій JAX-WS та JAX-RPC?
2. Які особливості має технологія JAX-WS?
3. Яке призначення має анотація @WebService.?
4. Укажіть особливості використання анотації @WebMethod.
5. У яких випадках застосовується анотація @WebParam?
6. Яке призначення має анотація @WebResult?
7. Укажіть особливості використання анотації @Oneway.
8. У яких випадках застосовується анотація @SOAPBinding?
9. Опишіть основні компоненти додатка на основі JAX-WS-технології.
10. Яким чином створюється клієнт для JAX-WS сервісу?

5 РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Машнин Т.С. Web-сервисы Java. – СПб.: БХВ-Петербург, 2012. – 560 с.
2. Hewitt E. Java SOA Cookbook. – Sebastopol, CA: O'Reilly Media, 2009. – 742 p.
3. Bean J. SOA and Web Services Interface Design: Principles, Techniques, and Standards. – Burlington, MA, USA: Elsevier Inc., 2010. – 360 p.