МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет информационных систем и технологий

Кафедра «Информационные системы»

Дисциплина «Интернет программирование»

**Отчёт по лабораторной работе №2**

**Инверсия управления и внедрение зависимостей**

Выполнил студент

гр. ПИбд-21:

Сергеев Н.И.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Научный руководитель:

Филиппов А.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ульяновск

2022

**Задание**

1. Реализовать интерфейс, в котором объявлено не менее 4 функций для выполнения операций над данными. Параметры и тип возвращаемого значения функций должны быть объявлены через обобщения (Generic).
2. Создать реализации интерфейса для типов String и Integer.
3. Реализовать контроллер (RestController), в котором содержится 4 метода (см. интерфейс). Параметры метода контроллера должны соответствовать параметрам аналогичного метода из интерфейса (возможно изменение типа данных на Object). Кроме того, каждый метод контроллера должен содержать параметр type с типом String. Параметр type используется для выбора реализации интерфейса: string или integer.
4. Реализовать класс с бизнес-логикой (Service) для контроллера. В зависимости от указанного значения параметра type необходимо получать ссылку на соответствующую реализацию интерфейса через механизм поиска зависимостей.
5. Контроллер должен использовать сервис через механизм внедрения зависимостей.
6. Написать модульные тесты для тестирования класса с бизнес-логикой.

**Сервер**

Файл IMethod<T> – параметризованный класс-интерфейс для 4 операций с объектом T.

Методы:

* Sum – сложение двух объектов;
* Minus – разность двух объектов;
* Multiply – произведение двух объектов;
* Div – деление двух объектов;

package ip.labwork.method.domain;

public interface IMethod<T> {

T Sum(T first, T second);

T Multiply(T first, Integer second);

T Minus(T first, Integer second);

T Div(T first, T second);

}

Файл MethodInt – файл-реализация интерфейса IMethod, параметр - Integer.

package ip.labwork.method.domain;

import org.springframework.stereotype.Component;

@Component(value="int")

public class MethodInt implements IMethod<Integer>{

public Integer Sum(Integer first, Integer second) {

return Integer.parseInt(first.toString()) + Integer.parseInt(second.toString());

}

public Integer Multiply(Integer first, Integer second) {

return Integer.parseInt(first.toString()) \* Integer.parseInt(second.toString());

}

public Integer Minus(Integer first, Integer second) {

return Integer.parseInt(first.toString()) - Integer.parseInt(second.toString());

}

public Integer Div(Integer first, Integer second) {

int num = Integer.parseInt(second.toString());

if (num == 0){

return null;

}else{

return Integer.parseInt(first.toString()) / num;

}

}

}

Файл MethodString – файл-реализация интерфейса IMethod, параметр - String.

package ip.labwork.method.domain;

import org.springframework.stereotype.Component;

@Component(value="string")

public class MethodString implements IMethod<String>{

@Override

public String Sum(String first, String second) {

return first.concat(second);

}

@Override

public String Multiply(String first, Integer second) {

if (second != 0){

String temp = "";

for (int i = 0; i < second; i++){

temp = temp.concat(first);

}

return temp;

}

else{

return first;

}

}

@Override

public String Minus(String first, Integer second) {

String temp = first;

if(temp.length() >= second){

return temp.substring(0, first.length() - second);

}else{

return first;

}

}

@Override

public String Div(String first, String second) {

if (first.contains(second)){

return "true";

}else{

return "false";

}

}

}

Файл MethodService –бизнес-логика.

package ip.labwork.method.service;

import ip.labwork.method.domain.IMethod;

import ip.labwork.method.domain.MethodString;

import org.springframework.context.ApplicationContext;

import org.springframework.stereotype.Service;

@Service

public class MethodService {

private final ApplicationContext applicationContext;

public MethodService(ApplicationContext applicationContext) {

this.applicationContext = applicationContext;

}

public String Sum(Object first, Object second, String type) {

final IMethod speaker = (IMethod) applicationContext.getBean(type);

if (speaker instanceof MethodString){

return String.format("%s", speaker.Sum(first,second));

}else{

return String.format("%s", speaker.Sum(Integer.parseInt(first.toString()),Integer.parseInt(second.toString())));

}

}

public String Ras(Object first, Object second, String type) {

final IMethod speaker = (IMethod) applicationContext.getBean(type);

if (speaker instanceof MethodString){

return String.format("%s", speaker.Minus(first,Integer.parseInt(second.toString())));

}else{

return String.format("%s", speaker.Minus(Integer.parseInt(first.toString()),Integer.parseInt(second.toString())));

}

}

public String Pros(Object first, Object second, String type) {

final IMethod speaker = (IMethod) applicationContext.getBean(type);

if (speaker instanceof MethodString){

return String.format("%s", speaker.Multiply(first,Integer.parseInt(second.toString())));

}else{

return String.format("%s", speaker.Multiply(Integer.parseInt(first.toString()),Integer.parseInt(second.toString())));

}

}

public String Del(Object first, Object second, String type) {

final IMethod speaker = (IMethod) applicationContext.getBean(type);

if (speaker instanceof MethodString){

return String.format("%s", speaker.Div(first,second));

}else {

return String.format("%s", speaker.Div(Integer.parseInt(first.toString()), Integer.parseInt(second.toString())));

}

}

}

Файл MethodController – REST-контроллер 4 операций.

package ip.labwork.method.controller;

import ip.labwork.method.service.MethodService;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class MethodController {

private final MethodService speakerService;

public MethodController(MethodService speakerService) {

this.speakerService = speakerService;

}

@GetMapping("/sum")

public String Sum(@RequestParam(value = "first", defaultValue = "1") Object first,

@RequestParam(value = "second", defaultValue = "1") Object second,

@RequestParam(value = "type", defaultValue = "int") String type) {

return speakerService.Sum(first,second,type);

}

@GetMapping("/minus")

public String Ras(@RequestParam(value = "first", defaultValue = "1") Object first,

@RequestParam(value = "second", defaultValue = "1") Object second,

@RequestParam(value = "type", defaultValue = "int") String type) {

return speakerService.Ras(first,second,type);

}

@GetMapping("/multi")

public String Pros(@RequestParam(value = "first", defaultValue = "1") Object first,

@RequestParam(value = "second", defaultValue = "1") Object second,

@RequestParam(value = "type", defaultValue = "int") String type) {

return speakerService.Pros(first,second,type);

}

@GetMapping("/div")

public String Del(@RequestParam(value = "first", defaultValue = "1") Object first,

@RequestParam(value = "second", defaultValue = "1") Object second,

@RequestParam(value = "type", defaultValue = "int") String type) {

return speakerService.Del(first,second,type);

}

}

**Клиент**

Файл index.html

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<script src="node\_modules/bootstrap/dist/js/bootstrap.min.js"></script>

<link href="node\_modules/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet"/>

<link href="node\_modules/@fortawesome/fontawesome-free/css/all.min.css" rel="stylesheet"/>

<title>Document</title>

</head>

<body>

<form id="frm-items" class="row g-3">

<div class="d-flex flex-column w-25 my-5 mx-3">

Введите первое число

<input id="first"></input>

Введите второе число

<input id="second" ></input>

Выберите операцию

<div>

<button type="button" class="btn btn-primary" id="buttonPlus">+</button>

<button type="button" class="btn btn-primary" id="buttonMinus">-</button>

<button type="button" class="btn btn-primary" id="buttonMulti">\*</button>

<button type="button" class="btn btn-primary" id="buttonDiv">/</button>

</div>

Выберите тип данных

<div>

<select id="type" class="form-select" >

<option value="int">Число</option>

<option value="string">Строка</option>

</select>

</div>

Результат

<input id="res" type="text"></input>

</div>

</form>

<script src="/script.js"></script>

</body>

</html>

Файл script.js

let calculateButton = document.getElementById("calculate");

let numberOneInput = document.getElementById("first");

let numberTwoInput = document.getElementById("second");

let resultInput = document.getElementById("res");

let typeInput = document.getElementById("type");

buttonPlus.onclick = function(event) {

event.preventDefault();

let num\_1 = numberOneInput.value;

let num\_2 = numberTwoInput.value;

let type = typeInput.value;

fetch(`http://localhost:8080/sum?first=${num\_1}&second=${num\_2}&type=${type}`)

.then(response => response.text())

.then(res => {

resultInput.value = res;

});

}

buttonMinus.onclick = function(event) {

event.preventDefault();

let num\_1 = numberOneInput.value;

let num\_2 = numberTwoInput.value;

let type = typeInput.value;

fetch(`http://localhost:8080/minus?first=${num\_1}&second=${num\_2}&type=${type}`)

.then(response => response.text())

.then(res => {

resultInput.value = res;

});

}

buttonMulti.onclick = function(event) {

event.preventDefault();

let num\_1 = numberOneInput.value;

let num\_2 = numberTwoInput.value;

let type = typeInput.value;

fetch(`http://localhost:8080/multi?first=${num\_1}&second=${num\_2}&type=${type}`)

.then(response => response.text())

.then(res => {

resultInput.value = res;

});

}

buttonDiv.onclick = function(event) {

event.preventDefault();

let num\_1 = numberOneInput.value;

let num\_2 = numberTwoInput.value;

let type = typeInput.value;

fetch(`http://localhost:8080/div?first=${num\_1}&second=${num\_2}&type=${type}`)

.then(response => response.text())

.then(res =>

{

resultInput.value = res;

});

}