ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. техн. наук, доцент |  |  |  | А. В. Фомин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| РЕАЛИЗАЦИЯ SOAP ВЕБ-СЕРВИСА |
| по курсу: СОВРЕМЕННЫЕ ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4232М |  |  |  | В. Ф. Губайдулин |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**Цель** **работы:**

Изучение средств Java для разработки web сервисов.

**Вариант:**

Разрабатываемое приложение ­– автоматизированная система управления составом футбольной команды.

Система будет позволять работать со списком команд, с игроками, играющими за определённую команду, а также с формациями – схемами игры.

**Ход работы:**

1) SOAP (Simple Object Access Protocol) — протокол обмена структурированными сообщениями в распределённой вычислительной среде. Сообщение SOAP выглядит так:

* Envelope — корневой элемент, который определяет сообщение и пространство имен, использованное в документе.
* Header — содержит атрибуты сообщения, например: информация о безопасности или о сетевой маршрутизации.
* Body — содержит сообщение, которым обмениваются приложения.
* Fault — необязательный элемент, который предоставляет информацию об ошибках, которые произошли при обработке сообщений.

WSDL (Web Services Description Language) — язык описания веб-сервисов и доступа к ним, основанный на языке XML.

2) Был реализован класс WSConfig, который определяет Endpoint’ы для сервисов.

Листинг 1 – Класс WSConfig

@Configuration

public class WSConfig {

@Autowired

private Bus bus;

@Autowired

private IPlayerRepos \_playerRepos;

@Autowired

private ISkillsRepos \_skillsRepos;

@Bean

public Endpoint helloEndpoint() {

EndpointImpl endpoint = new EndpointImpl(bus, new HelloServiceImpl());

endpoint.publish("/Hello");

return endpoint;

}

@Bean

public Endpoint addSkillsEndpoint() {

EndpointImpl endpoint = new EndpointImpl(bus, new SkillsServiceImpl(\_playerRepos, \_skillsRepos));

endpoint.publish("/Skills");

return endpoint;

}

}

3) Были реализованы сервисы и их интерфейсы: HelloService, SkillsService.

Реализация класса SkillsServiceImpl представлен в приложении А. В классе SkillsServiceImpl реализованы следующие методы:

* addOrUpdSkills(SkillsDto skills) – реализует добавление или обновление критериев навыков для игроков в зависимости от года начала сезона.
* getSkills(Integer playerId, Integer season) – реализует получение критериев навыков игрока в зависимости от года начала сезона.
* delSkills(Integer playerId, Integer season) – реализует удаление критериев навыков для игроков в зависимости от года начала сезона.

Листинг 2 – Реализация SkillsService

@WebService(name = "addSkill")

public interface SkillsService {

@WebMethod(operationName = "addSkill")

String addOrUpdSkills(@WebParam(name="skills", targetNamespace = "") SkillsDto skills) throws SoapException;

@WebMethod(operationName = "getSkill")

Skills getSkills(@WebParam(name="playerId", targetNamespace = "") Integer playerId,

@WebParam(name="season", targetNamespace = "") Integer season) throws SoapException;

@WebMethod(operationName = "delSkill")

String delSkills(@WebParam(name="playerId", targetNamespace = "") Integer playerId,

@WebParam(name="season", targetNamespace = "") Integer season) throws SoapException;

}

4) Вышеописанные методы были протестированы с помощью SoapUI.

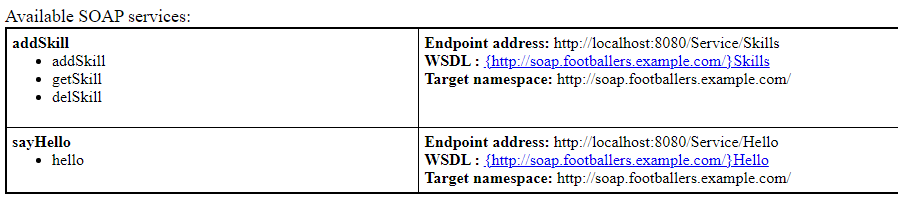
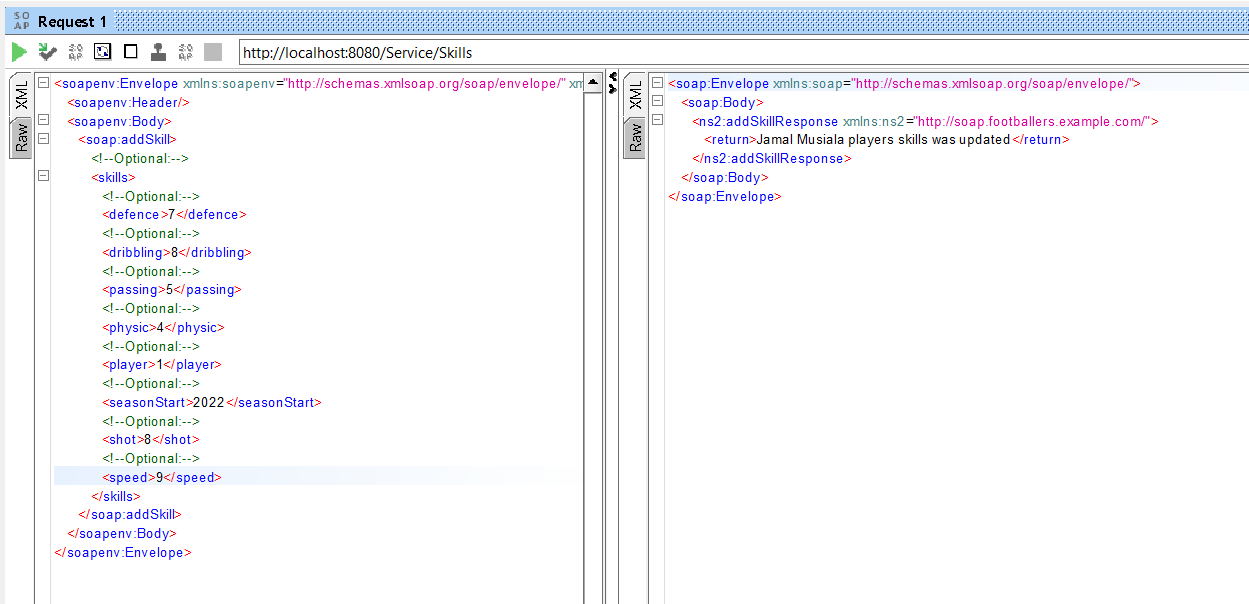


Рисунок 1 – Список доступных SOAP сервисов



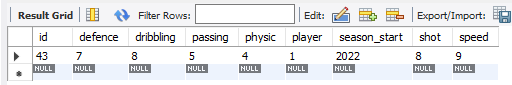


Рисунок 2 – Тест метода addOrUpdSkills

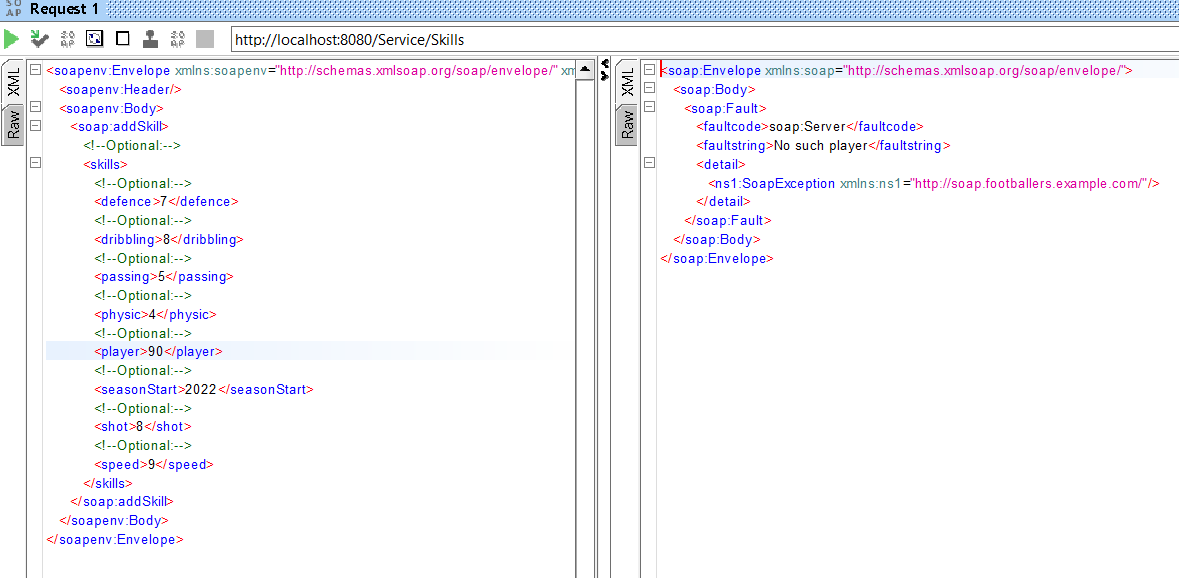


Рисунок 3 – Тест метода addOrUpdSkills если нет игрока с введённым id

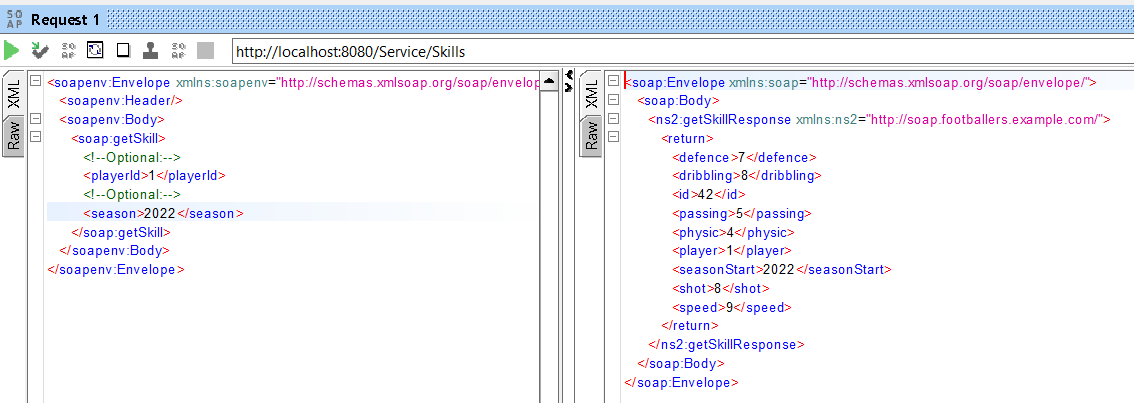


Рисунок 4 – Тест метода getSkills

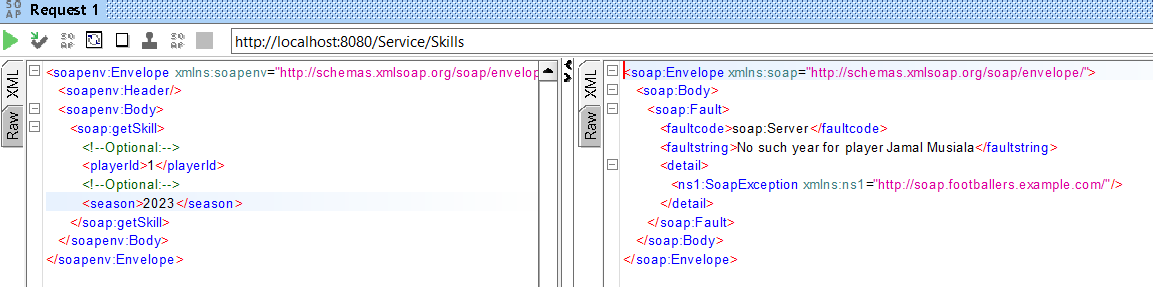


Рисунок 5 – Тест метода getSkills при неверно введённом годе

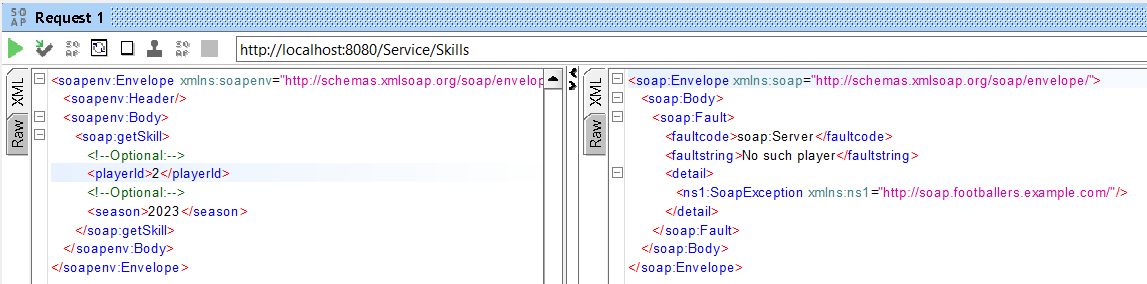


Рисунок 6 – Тест метода getSkills при неверно введённом id игрока

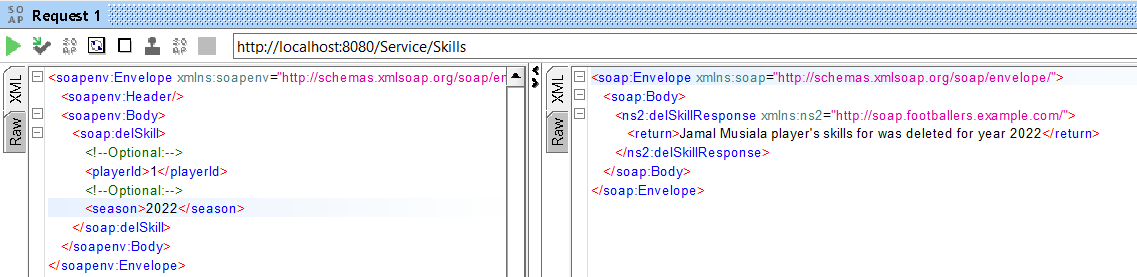


Рисунок 7 – Тест метода delSkills

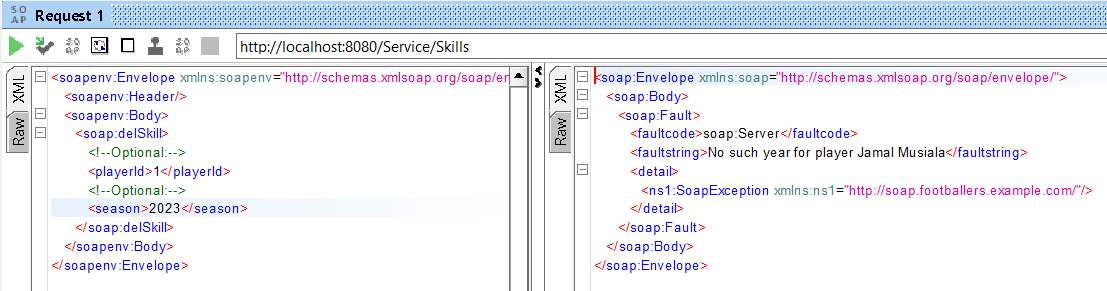


Рисунок 8 – Тест метода delSkills при неверно введённом годе

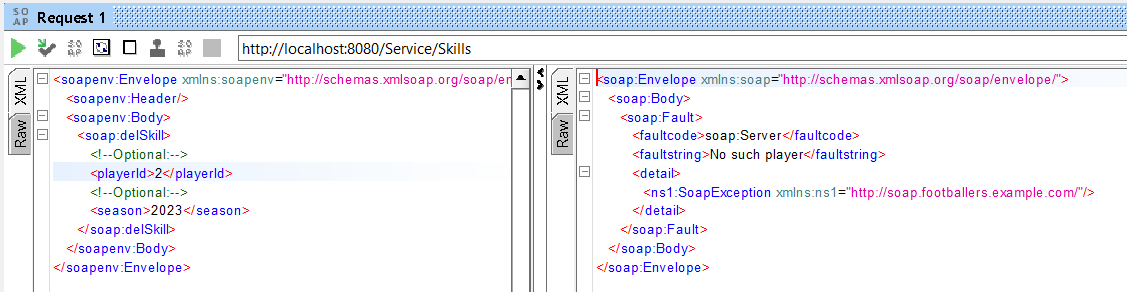


Рисунок 9 – Тест метода delSkills при неверно введённом id игрока

**Вывод:**

Были изучены средства Java для разработки web сервисов. Были реализованы и протестированы методы, использующиеся web сервисом.

**Приложение А. Листинг класса SkillsServiceImpl**

@WebService(serviceName = "Skills")

public class SkillsServiceImpl implements SkillsService {

private final IPlayerRepos \_playerRepos;

private final ISkillsRepos \_skillsRepos;

public SkillsServiceImpl(IPlayerRepos playerRepos, ISkillsRepos skillsRepos){

\_playerRepos = playerRepos;

\_skillsRepos = skillsRepos;

}

@Override

public String addOrUpdSkills(SkillsDto skills) throws SoapException {

String result = "{0} player's skills was updated";

Players player = \_playerRepos.findById(skills.getPlayer()).orElse(null);

if (player == null) {

Throwable t = new NoSuchElementException("No such player");

throw new SoapException("No such player", t);

}

Skills existedSkills = \_skillsRepos.getSkillsByPlayerAndSeason(skills.getPlayer(), skills.getSeasonStart());

if (existedSkills == null) {

Skills skillsToAdd = SkillsMapper.INSTANCE.toModel(skills);

\_skillsRepos.save(skillsToAdd);

} else {

if (skills.getDefence() != null)

existedSkills.setDefence(skills.getDefence());

if (skills.getPassing() != null)

existedSkills.setPassing(skills.getPassing());

if (skills.getDribbling() != null)

existedSkills.setDribbling(skills.getDribbling());

if (skills.getPhysic() != null)

existedSkills.setPhysic(skills.getPhysic());

if (skills.getShot() != null)

existedSkills.setShot(skills.getShot());

if (skills.getSpeed() != null)

existedSkills.setSpeed(skills.getSpeed());

\_skillsRepos.save(existedSkills);

}

return MessageFormat.format(result, player.getName());

}

@Override

public Skills getSkills(Integer playerId, Integer season) throws SoapException {

Players player = \_playerRepos.findById(playerId).orElse(null);

if (player == null) {

Throwable t = new NoSuchElementException("No such player");

throw new SoapException("No such player", t);

}

Skills existedSkills = \_skillsRepos.getSkillsByPlayerAndSeason(playerId, season);

if (existedSkills == null) {

Throwable t = new NoSuchElementException("No such year for player " + player.getName());

throw new SoapException("No such year for player " + player.getName(), t);

}

return existedSkills;

}

@Override

public String delSkills(Integer playerId, Integer season) throws SoapException {

String result = "%s player's skills for was deleted for year %s";

Players player = \_playerRepos.findById(playerId).orElse(null);

if (player == null) {

Throwable t = new NoSuchElementException("No such player");

throw new SoapException("No such player", t);

}

Skills existedSkills = \_skillsRepos.getSkillsByPlayerAndSeason(playerId, season);

if (existedSkills == null) {

Throwable t = new NoSuchElementException("No such year for player " + player.getName());

throw new SoapException("No such year for player " + player.getName(), t);

}

\_skillsRepos.delete(existedSkills);

return String.format(result, player.getName(), season.toString());

}

}