**Министерство образования и науки Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**“НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**УНИВЕРСИТЕТ ИТМО**”

**Факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА № 2**

**ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИИ ВЕБ-СЕРВИСОВ»**

Студент: Норин Евгений Рустамович

Группа: P41142

Преподаватель: Дергачев Андрей Михайлович

Санкт-Петербург

2020

**Задание:**

В данной работе в веб-сервис, разработанный в первой работе, необходимо добавить методы для создания, изменения и удаления записей из таблицы. Метод создания должен принимать значения полей новой записи, метод изменения – идентификатор изменяемой записи, а также новые значения полей, а метод удаления – только идентификатор удаляемой записи. Метод создания должен возвращать идентификатор новой записи, а методы обновления или удаления – статус операции. В данной работе следует вносить изменения только в standalone-реализацию сервиса. В соответствии с изменениями сервиса необходимо обновить и клиентское приложение.

**Выполнение работы:**

В ходе данной лабораторной работы был дополнен код *ArticlesDao.java*(листинг 1.1)и *ArticlesWebService.java*(листинг 1.2)методами *createArticle*, *updateArticle* и *deleteArticle*, и *QueryBuilder.java*(листинг 1.3)методом *buildUpdate.*

Листинг 1.1 *ArticlesDao.java*

**package** ru.itmo.webservices.secondlab.standalone.dao;  
  
**import** ru.itmo.webservices.secondlab.standalone.pojo.Article;  
  
**import** java.sql.\*;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.logging.Level;  
**import** java.util.logging.Logger;  
  
**public class** ArticlesDao {  
 **private** Connection **connection**;  
  
 **public** ArticlesDao() {  
 **connection** = ConnectionUtil.*getConnection*();  
 }  
  
 **public** List<Article> select(String authorId, Long hIndex, String articleName, String articleDesc, Long dateAdded) {  
 List<Article> articles = **new** ArrayList<>();  
 **try** (Connection connection = ConnectionUtil.*getConnection*()) {  
 Statement stmt = connection.createStatement();  
 ResultSet rs = stmt.executeQuery(  
 QueryBuilder.*buildSelect*(  
 authorId,  
 hIndex,  
 articleName,  
 articleDesc,  
 dateAdded  
 )  
 );  
  
 **while** (rs.next()) {  
 articles.add(  
 **new** Article(  
 rs.getString(**"author\_id"**),  
 rs.getLong(**"h\_index"**),  
 rs.getString(**"article\_name"**),  
 rs.getString(**"article\_desc"**),  
 rs.getLong(**"date\_added"**)  
 )  
 );  
 }  
 } **catch** (SQLException ex) {  
 Logger.*getLogger*(ArticlesDao.**class**.getName()).log(Level.***SEVERE***, **null**, ex);  
 }  
  
 **return** articles;  
 }  
  
 **public long** insert(String authorId, Long hIndex, String articleName, String articleDesc, Long dateAdded) {  
 String sql = **"INSERT INTO Article (author\_id, h\_index, article\_name, article\_desc, date\_added) VALUES (?, ?, ?, ?, ?)"**;  
 **try** {  
 PreparedStatement preparedStatement = **this**.**connection**.prepareStatement(sql, Statement.***RETURN\_GENERATED\_KEYS***);  
 preparedStatement.setString(1, authorId);  
 preparedStatement.setLong(2, hIndex);  
 preparedStatement.setString(3, articleName);  
 preparedStatement.setString(4, articleDesc);  
 preparedStatement.setLong(5, dateAdded);  
  
 **int** affectedRows = preparedStatement.executeUpdate();  
  
 **if** (affectedRows == 0) {  
 **return** -1;  
 }  
  
 ResultSet generatedKeys = preparedStatement.getGeneratedKeys();  
 **if** (generatedKeys.next()) {  
 **return** generatedKeys.getLong(1);  
 } **else** {  
 **return** -1;  
 }  
 } **catch** (SQLException e) {  
 **return** -1;  
 }  
 }  
  
 **public int** update(**long** id, String authorId, Long hIndex, String articleName, String articleDesc, Long dateAdded) {  
 String sql = QueryBuilder.*buildUpdate*(id, authorId, hIndex, articleName, articleDesc, dateAdded);  
 **try** {  
 PreparedStatement preparedStatement = **this**.**connection**.prepareStatement(sql);  
 **int** affectedRows = preparedStatement.executeUpdate();  
 **return** affectedRows == 0 ? -1 : 1;  
 } **catch** (SQLException e) {  
 **return** -1;  
 }  
 }  
  
 **public int** delete(**int** id) {  
 String sql = **"DELETE FROM article WHERE article\_id = ?"**;  
 **try** {  
 PreparedStatement preparedStatement = **this**.**connection**.prepareStatement(sql);  
 preparedStatement.setInt(1, id);  
  
 **int** affectedRows = preparedStatement.executeUpdate();  
 **return** affectedRows == 0 ? -1 : 1;  
 } **catch** (SQLException e) {  
 **return** -1;  
 }  
 }  
}

Листинг 1.2 *ArticlesWebService.java*

**package** ru.itmo.webservices.secondlab.standalone;  
  
**import** ru.itmo.webservices.secondlab.standalone.dao.ArticlesDao;  
**import** ru.itmo.webservices.secondlab.standalone.pojo.Article;  
  
**import** javax.jws.WebMethod;  
**import** javax.jws.WebParam;  
**import** javax.jws.WebService;  
**import** java.sql.SQLException;  
**import** java.util.List;  
  
@WebService(serviceName = **"ArticlesService"**)  
**public class** ArticleWebService {  
 @WebMethod  
 **public** List<Article> getArticles(@WebParam(name = **"authorId"**) String authorId,  
 @WebParam(name = **"hIndex"**) Long hIndex,  
 @WebParam(name = **"articleName"**) String articleName,  
 @WebParam(name = **"articleDesc"**) String articleDesc,  
 @WebParam(name = **"dateAdded"**) Long dateAdded) {  
 ArticlesDao dao = **new** ArticlesDao();  
 **return** dao.select(authorId, hIndex, articleName, articleDesc, dateAdded);  
 }  
  
 @WebMethod  
 **public long** createArticle(@WebParam(name = **"authorId"**) String authorId,  
 @WebParam(name = **"hIndex"**) Long hIndex,  
 @WebParam(name = **"articleName"**) String articleName,  
 @WebParam(name = **"articleDesc"**) String articleDesc,  
 @WebParam(name = **"dateAdded"**) Long dateAdded) {  
 ArticlesDao dao = **new** ArticlesDao();  
 **return** dao.insert(authorId, hIndex, articleName, articleDesc, dateAdded);  
 }  
  
 @WebMethod  
 **public long** updateArticle(@WebParam(name = **"id"**) Long id,  
 @WebParam(name = **"authorId"**) String authorId,  
 @WebParam(name = **"hIndex"**) Long hIndex,  
 @WebParam(name = **"articleName"**) String articleName,  
 @WebParam(name = **"articleDesc"**) String articleDesc,  
 @WebParam(name = **"dateAdded"**) Long dateAdded) {  
 ArticlesDao dao = **new** ArticlesDao();  
 **return** dao.update(id, authorId, hIndex, articleName, articleDesc, dateAdded);  
 }  
  
 @WebMethod  
 **public int** deleteArticle(@WebParam(name = **"id"**) **int** id) {  
 ArticlesDao dao = **new** ArticlesDao();  
 **return** dao.delete(id);  
 }  
}

Листинг 1.3 *QueryBuilder.java*

**package** ru.itmo.webservices.secondlab.standalone.dao;  
  
**public class** QueryBuilder {  
 **static** String buildSelect(String authorId, Long hIndex, String articleName, String articleDesc, Long dateAdded) {  
 StringBuilder builder = **new** StringBuilder(0);  
 **if** (authorId != **null**) {  
 builder.append(String.*format*(**"author\_id='%s'"**, authorId));  
 }  
 **if** (hIndex != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**" and h\_index=%d"**, hIndex));  
 } **else** {  
 builder.append(String.*format*(**"h\_index=%d"**, hIndex));  
 }  
 }  
 **if** (articleName != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**" and article\_name='%s'"**, articleName));  
 } **else** {  
 builder.append(String.*format*(**"article\_name='%s'"**, articleName));  
 }  
 }  
 **if** (articleDesc != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**" and article\_desc='%s'"**, articleDesc));  
 } **else** {  
 builder.append(String.*format*(**"article\_desc='%s'"**, articleDesc));  
 }  
 }  
 **if** (dateAdded != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**" and date\_added=%d"**, dateAdded));  
 } **else** {  
 builder.append(String.*format*(**"date\_added=%d"**, dateAdded));  
 }  
 }  
 **if** (builder.length() != 0) {  
 **return "select \* from article where "** + builder.toString();  
 } **else** {  
 **return "select \* from article;"**;  
 }  
 }  
  
 **static** String buildUpdate(Long id, String authorId, Long hIndex, String articleName, String articleDesc, Long dateAdded) {  
 StringBuilder builder = **new** StringBuilder(0);  
 **if** (authorId != **null**) {  
 builder.append(String.*format*(**"author\_id='%s'"**, authorId));  
 }  
 **if** (hIndex != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**",h\_index=%d"**, hIndex));  
 } **else** {  
 builder.append(String.*format*(**"h\_index=%d"**, hIndex));  
 }  
 }  
 **if** (articleName != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**",article\_name='%s'"**, articleName));  
 } **else** {  
 builder.append(String.*format*(**"article\_name='%s'"**, articleName));  
 }  
 }  
 **if** (articleDesc != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**",article\_desc='%s'"**, articleDesc));  
 } **else** {  
 builder.append(String.*format*(**"article\_desc='%s'"**, articleDesc));  
 }  
 }  
 **if** (dateAdded != **null**) {  
 **if** (builder.length() != 0) {  
 builder.append(String.*format*(**",date\_added=%d"**, dateAdded));  
 } **else** {  
 builder.append(String.*format*(**"date\_added=%d"**, dateAdded));  
 }  
 }  
 **if** (builder.length() != 0) {  
 **return "update article SET "** + builder.toString() + **" where article\_id="** + id.toString() + **";"**;  
 } **else** {  
 **return null**;  
 }  
 }  
}

WSDL сервиса по адресу <http://localhost:8080/ArticleService?wsdl> приведен в листинге 1.5

Листинг 1.5 WSDL сервиса

<?xml version="1.0" encoding="UTF-8"?> <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://standalone.secondlab.webservices.itmo.ru/" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata" xmlns:wsp="http://www.w3.org/ns/ws-policy" xmlns:wsp1\_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://standalone.secondlab.webservices.itmo.ru/" name="ArticlesService"> <types> <xsd:schema> <xsd:import namespace="http://standalone.secondlab.webservices.itmo.ru/" schemaLocation="http://localhost:8080/ArticleService?xsd=1" /> </xsd:schema> </types> <message name="getArticles"> <part name="parameters" element="tns:getArticles" /> </message> <message name="getArticlesResponse"> <part name="parameters" element="tns:getArticlesResponse" /> </message> <message name="createArticle"> <part name="parameters" element="tns:createArticle" /> </message> <message name="createArticleResponse"> <part name="parameters" element="tns:createArticleResponse" /> </message> <message name="updateArticle"> <part name="parameters" element="tns:updateArticle" /> </message> <message name="updateArticleResponse"> <part name="parameters" element="tns:updateArticleResponse" /> </message> <message name="deleteArticle"> <part name="parameters" element="tns:deleteArticle" /> </message> <message name="deleteArticleResponse"> <part name="parameters" element="tns:deleteArticleResponse" /> </message> <portType name="ArticleWebService"> <operation name="getArticles"> <input wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/getArticlesRequest" message="tns:getArticles" /> <output wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/getArticlesResponse" message="tns:getArticlesResponse" /> </operation> <operation name="createArticle"> <input wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/createArticleRequest" message="tns:createArticle" /> <output wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/createArticleResponse" message="tns:createArticleResponse" /> </operation> <operation name="updateArticle"> <input wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/updateArticleRequest" message="tns:updateArticle" /> <output wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/updateArticleResponse" message="tns:updateArticleResponse" /> </operation> <operation name="deleteArticle"> <input wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/deleteArticleRequest" message="tns:deleteArticle" /> <output wsam:Action="http://standalone.secondlab.webservices.itmo.ru/ArticleWebService/deleteArticleResponse" message="tns:deleteArticleResponse" /> </operation> </portType> <binding name="ArticleWebServicePortBinding" type="tns:ArticleWebService"> <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document" /> <operation name="getArticles"> <soap:operation soapAction="" /> <input> <soap:body use="literal" /> </input> <output> <soap:body use="literal" /> </output> </operation> <operation name="createArticle"> <soap:operation soapAction="" /> <input> <soap:body use="literal" /> </input> <output> <soap:body use="literal" /> </output> </operation> <operation name="updateArticle"> <soap:operation soapAction="" /> <input> <soap:body use="literal" /> </input> <output> <soap:body use="literal" /> </output> </operation> <operation name="deleteArticle"> <soap:operation soapAction="" /> <input> <soap:body use="literal" /> </input> <output> <soap:body use="literal" /> </output> </operation> </binding> <service name="ArticlesService"> <port name="ArticleWebServicePort" binding="tns:ArticleWebServicePortBinding"> <soap:address location="http://localhost:8080/ArticleService" /> </port> </service> </definitions>

Клиентский код сгенерирован командой:

~/IdeaProjects/webservices-labs(master) » wsimport -Xnocompile http://localhost:8080/ArticleService\?wsdl -d /Users/e.norin/IdeaProjects/webservices-labs/secondlab-client/src/main/java

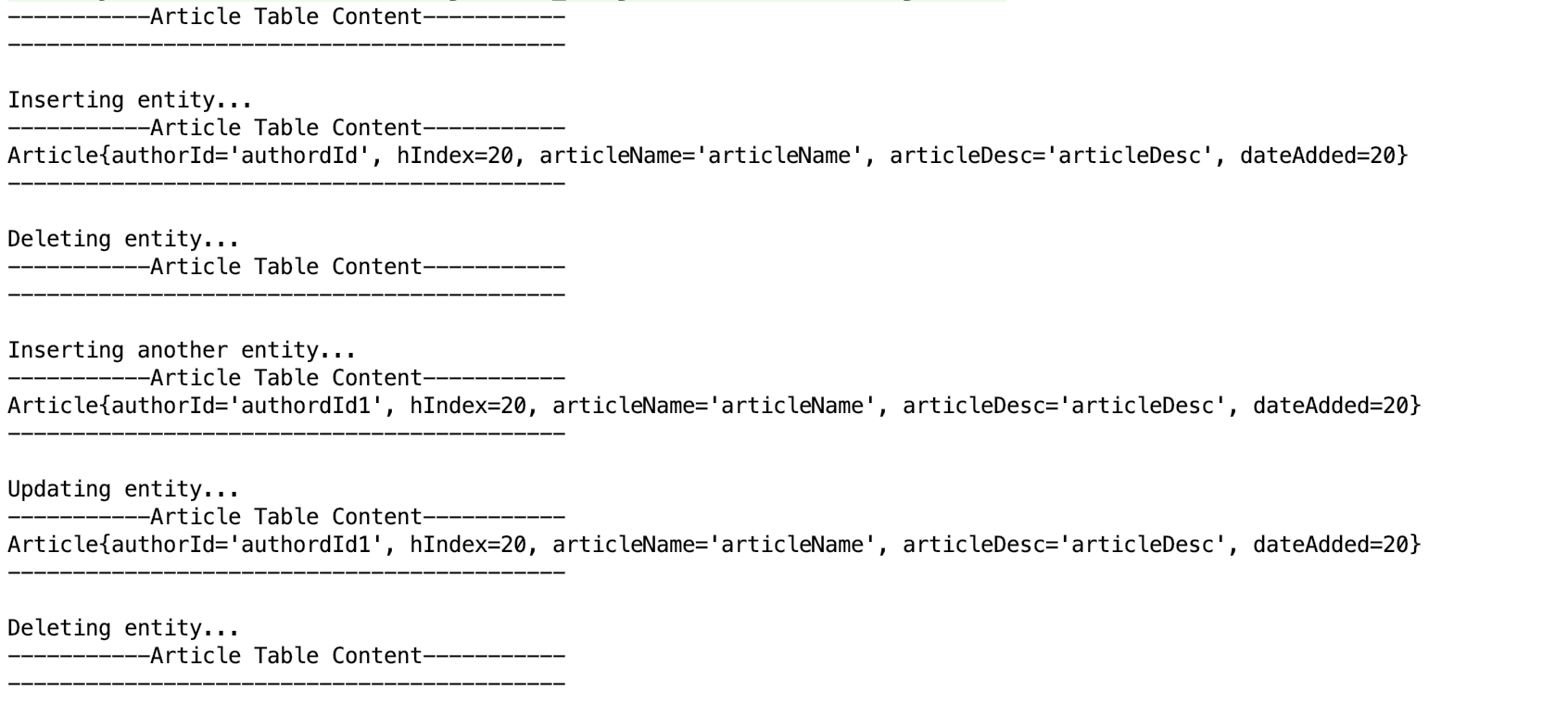
*WebServiceClient.java* (Листинг 1.6) создан для демонстрационных целей.

Листинг 1.6 *WebServiceClient.java*

**package** ru.itmo.webservices.secondlab.standalone.client;  
  
**import** ru.itmo.webservices.secondlab.standalone.Article;  
**import** ru.itmo.webservices.secondlab.standalone.ArticleWebService;  
**import** ru.itmo.webservices.secondlab.standalone.ArticlesService;  
  
**import** java.net.MalformedURLException;  
**import** java.net.URL;  
**import** java.util.List;  
  
**public class** WebServiceClient {  
 **public static void** getStatus(ArticleWebService articleWebService) {  
 System.***out***.println(**"Articles Status"**);  
 List<Article> articles = articleWebService.getArticles(**null**, **null**, **null**, **null**, **null**);  
 **for** (Article article : articles) {  
 System.***out***.println(article.toString());  
 }  
  
 System.***out***.println(**"Total articles: "** + articles.size());  
 System.***out***.println();  
 }  
  
 **public static void** main(String[] args) **throws** MalformedURLException {  
 URL url = **new** URL(**"http://localhost:8080/ArticleService?wsdl"**);  
 ArticlesService articlesService = **new** ArticlesService(url);  
 ArticleWebService articleWebService = articlesService.getArticleWebServicePort();  
 *getStatus*(articleWebService);  
  
 System.***out***.println(**"Inserting entity..."**);  
 **int** id = (**int**)articleWebService.createArticle(**"authordId"**, 20L, **"articleName"**, **"articleDesc"**, 20L);  
 *getStatus*(articleWebService);  
  
 System.***out***.println(**"Deleting entity..."**);  
 articleWebService.deleteArticle(id);  
 *getStatus*(articleWebService);  
  
 System.***out***.println(**"Inserting another entity..."**);  
 **long** id1 = articleWebService.createArticle(**"authordId1"**, 20L, **"articleName"**, **"articleDesc"**, 20L);  
 *getStatus*(articleWebService);  
  
 System.***out***.println(**"Updating entity..."**);  
 articleWebService.updateArticle(id1, **"authordId1"**, 20L, **"articleName"**, **"articleDesc"**, 20L);  
 *getStatus*(articleWebService);  
  
 System.***out***.println(**"Deleting entity..."**);  
 articleWebService.deleteArticle((**int**)id1);  
 *getStatus*(articleWebService);  
 }  
}

Результат выполнения клиентского приложения представлен на Рисунке 1.1

Рисунок 1.1 Запуск клиентского приложения



**Вывод**: в ходе выполнения работы был реализован CRUD с помощью SOAP-сервиса в виде standalone-приложения. Для демонстрации работы разработанного сервиса было разработано клиентское консольное приложение.