МИНИТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К курсовой работе

По курсу «Программирование на языке Java»

На тему «Разработка многомодульного приложения на языке Java»

Выполнил:

Студент группы 20ВВВ1

Тюгаев К.А.

Приняла:

к.т.н., доцент Юрова О.В.

к.т.н., доцент Карамышева Н.С.

Пенза 2023

**Оглавление**

[Введение 3](#_Toc104748500)

[1. Постановка задачи 6](#_Toc104748501)

[2. Выбор решения 7](#_Toc104748502)

[3. Описание программы 8](#_Toc104748503)

[4. Экспериментальная часть 10](#_Toc104748504)

[Заключение 13](#_Toc104748505)

[Список используемой литературы. 14](#_Toc104748506)

[Приложение A.2 – ServerSomething.java 18](#_Toc104748507)

[Приложение Б. Исходный код клиентской части 22](#_Toc104748508)

[Приложение Б.1 – MainframeClient.java 22](#_Toc104748509)

[Приложение В. UML-диаграммы приложения 40](#_Toc104748510)

# Введение

Java – один из самых популярных языков программирования, с большим количеством достоинств:

* Язык Java прост для изучения.

При разработке Java было уделено большое внимание простоте языка, поэтому программы на Java, по сравнению с программами на других языках, проще писать, компилировать, отлаживать и изучать.

* Java - это объектно-ориентированный язык.

Это позволяет создавать модульные программы, исходный код которых может использоваться многократно.

* Язык Java не зависит от платформы.

Одним из основных преимуществ языка Java является возможность переноса программ из одной системы в другую. Поскольку программы на Java не зависят от платформы как на уровне исходного кода, так и на двоичном уровне, их можно запускать в различных системах, что особенно важно для программ, предназначенных для World Wide Web.

.

# Постановка задачи

Разработать клиент-серверное приложение, реализующее чат

Функции сервера:

1) Обработка данных от пользователей.

Функции клиента:

1) Реализация интерфейса, и его изменение в соответствии с ответами сервера.

Приложение должно обладать графическим интерфейсом и использовать следующие технологии:

1. Java Collections Framework
2. Механизм обработки исключительных ситуаций
3. Java Stream API
4. Java Multithreading
5. Сетевое взаимодействие.

ОС – Windows. Язык программирования: Java. Среда разработки NetBeans.

# 2. Выбор решения

WebSocket — это независимый протокол, основанный на протоколе TCP и предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени. Таким образом, изначально синхронный протокол HTTP, построенный на модели «запрос — ответ», становится полностью асинхронным и симметричным. При использовании WebSocket'a нет клиента и сервера с фиксированными ролями, а есть два равноправных участника обмена данными. Каждый участник функционирует самостоятельно : отправил сообщение и продолжил выполнять свои функции. Участник, получивший сообщение, может вообще не отвечать : протокол дает полную свободу в обмене данными.

# 3. Описание программы

Программа, состоит из клиентской и серверной части. Сначала запускается сервер, который входит в состояние ожидания подключения клиентов.

На других устройствах запускается клиентская часть, которая по нажатию определенной кнопки, пытается подключиться к серверу используя WebSocket.

# 

# 4. Экспериментальная часть

Для начала запускаем сервер.

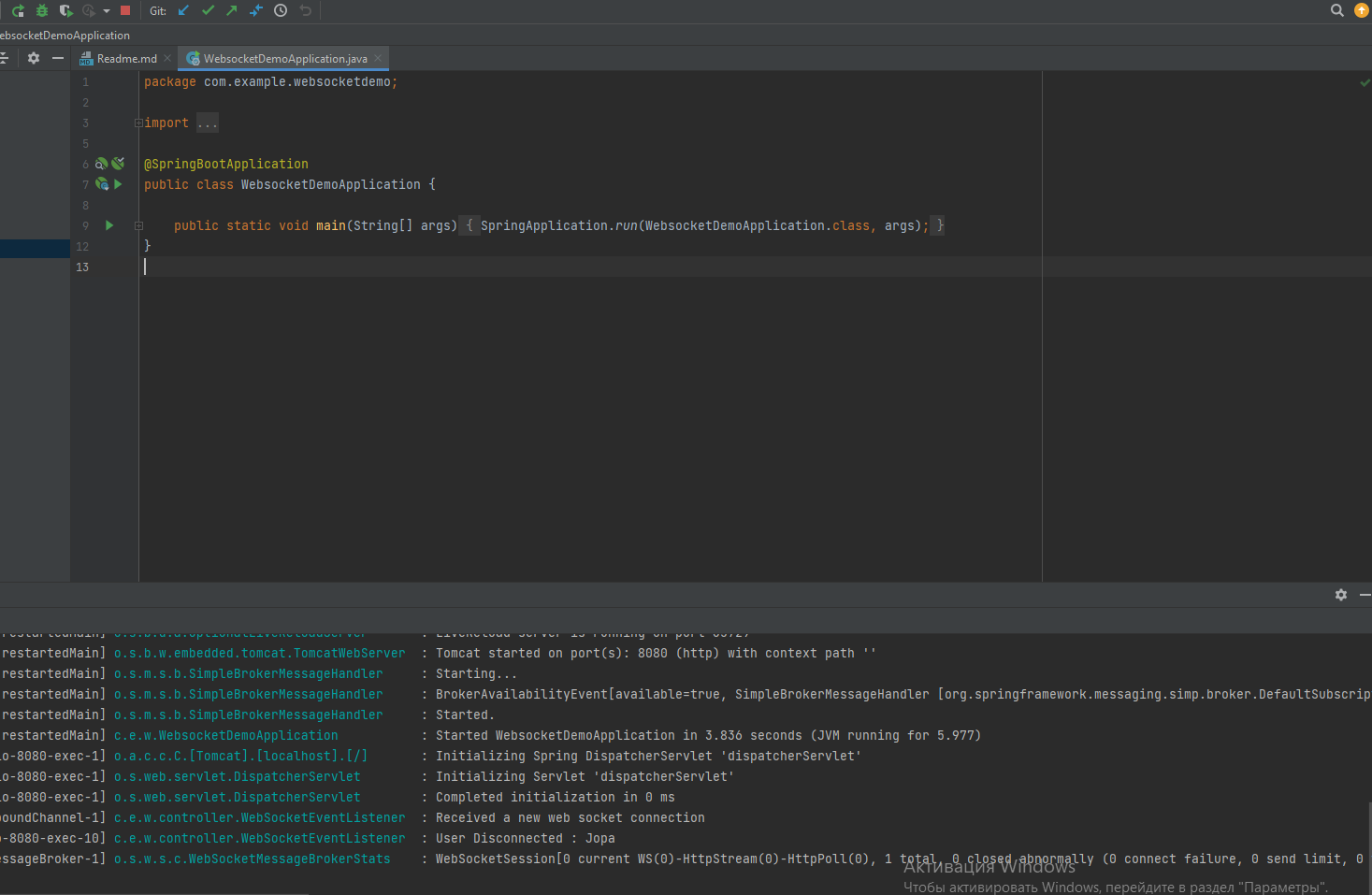


Рисунок – 1 Запуск сервера.

Далее запускаем клиента.

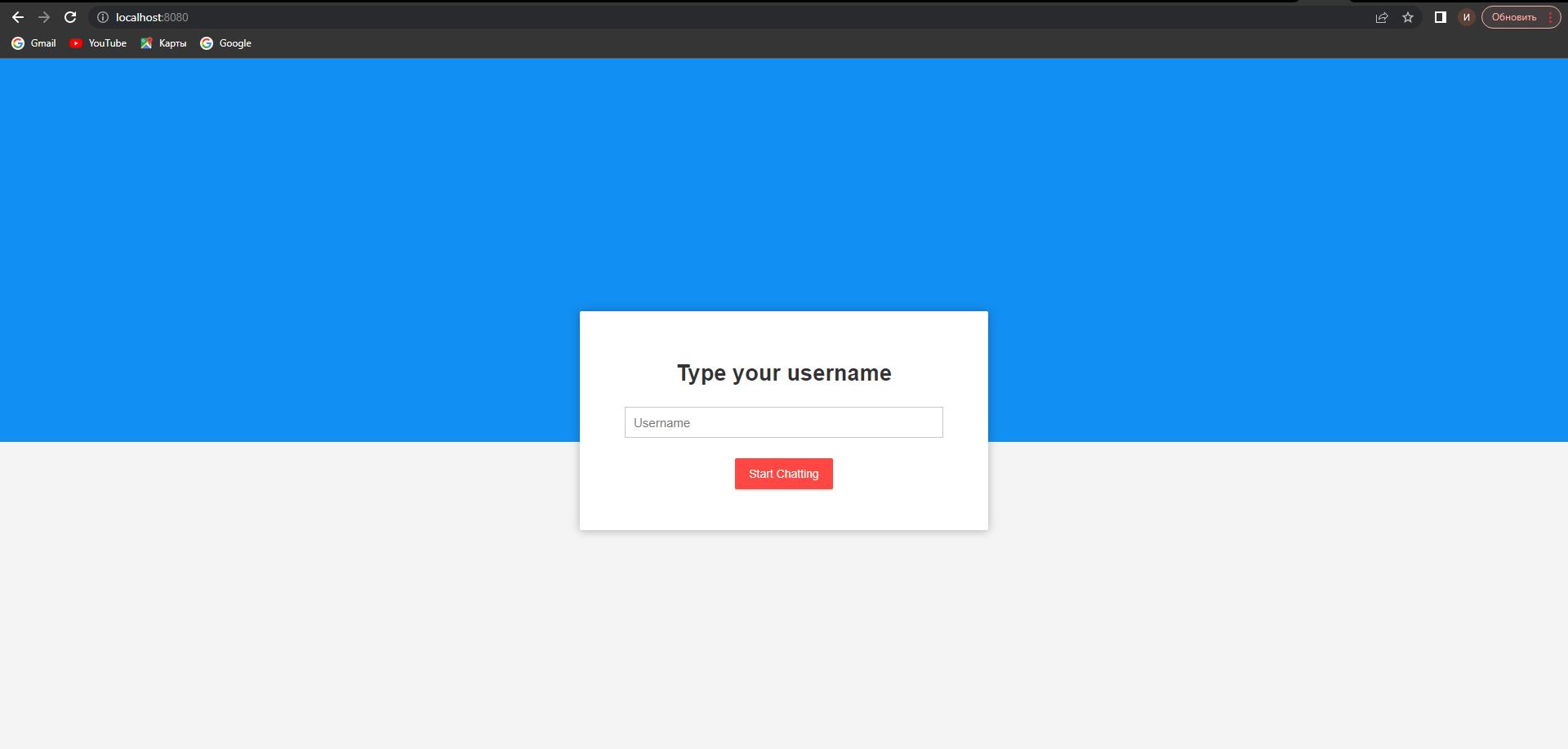


Рисунок – 2 Запуск Клиента.

После ввода имени и нажатия на кнопку “Start Chating” открывается страница с полем для вывода сообщений пользователей

# Заключение

При выполнении данной курсовой работы были получены навыки разработки программ на языке высокого уровня Java. Были освоены навыки разработки с использованием фреймворка Spring. Изучены технологии WebSocket. Освоены навыки разработки клиент-серверного приложения.

В рамках данной работы было написано приложение «Чат».

# Список используемой литературы.

# 1. “Spring 5 для профессионалов” | Хо Кларенс, Шефер Крис

2. Мейер Б. Объектно-ориентированное программирование и программная инженерия / Мейер Б.— Электрон. текстовые данные.— М.: Интернет Университет Информационных Технологий (ИНТУИТ), 2016.— 285с.

3.Java. Руководство для начинающих - Герберт Шилдт

##### **Приложение А. Исходный код серверной части.**

##### **А.1 – ChatMessage.java**

package com.example.websocketdemo.model;  
  
  
public class ChatMessage {  
 private MessageType type;  
 private String content;  
 private String sender;  
  
 public enum MessageType {  
 CHAT,  
 JOIN,  
 LEAVE  
 }  
  
 public MessageType getType() {  
 return type;  
 }  
  
 public void setType(MessageType type) {  
 this.type = type;  
 }  
  
 public String getContent() {  
 return content;  
 }  
  
 public void setContent(String content) {  
 this.content = content;  
 }  
  
 public String getSender() {  
 return sender;  
 }  
  
 public void setSender(String sender) {  
 this.sender = sender;  
 }  
}

## Приложение A.2 – WebSocketConfig.java

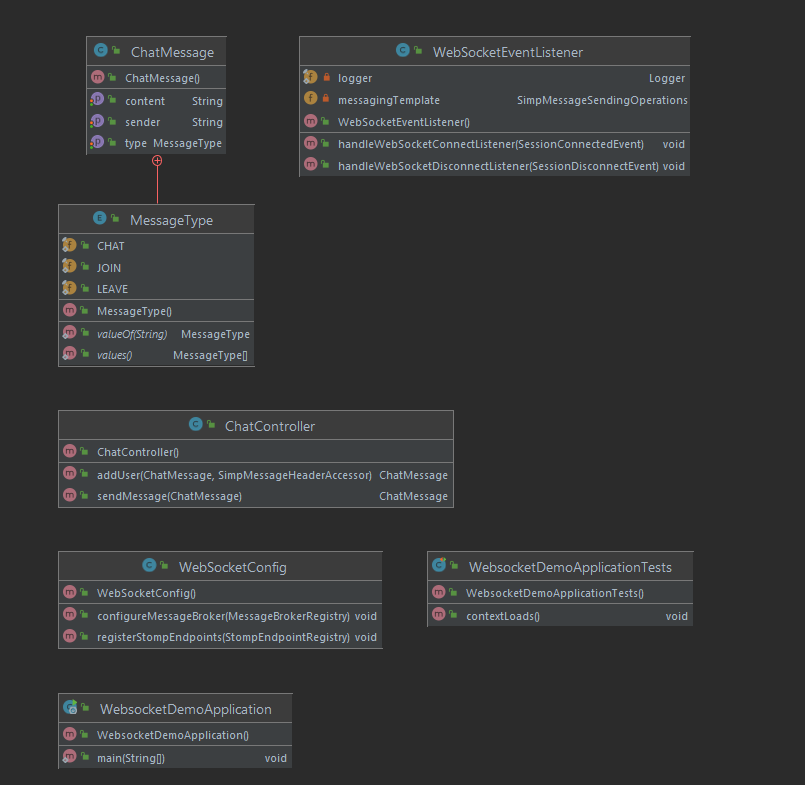
package com.example.websocketdemo.config;  
  
import org.springframework.context.annotation.Configuration;  
import org.springframework.messaging.simp.config.MessageBrokerRegistry;  
import org.springframework.web.socket.config.annotation.\*;  
  
  
@Configuration  
@EnableWebSocketMessageBroker  
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {  
  
 @Override  
 public void registerStompEndpoints(StompEndpointRegistry registry) {  
 registry.addEndpoint("/ws").withSockJS();  
 }  
  
 @Override  
 public void configureMessageBroker(MessageBrokerRegistry registry) {  
 registry.setApplicationDestinationPrefixes("/app");  
 registry.enableSimpleBroker("/topic"); // Enables a simple in-memory broker  
  
  
 }  
}

# Приложение Б. Исходный код клиентской части

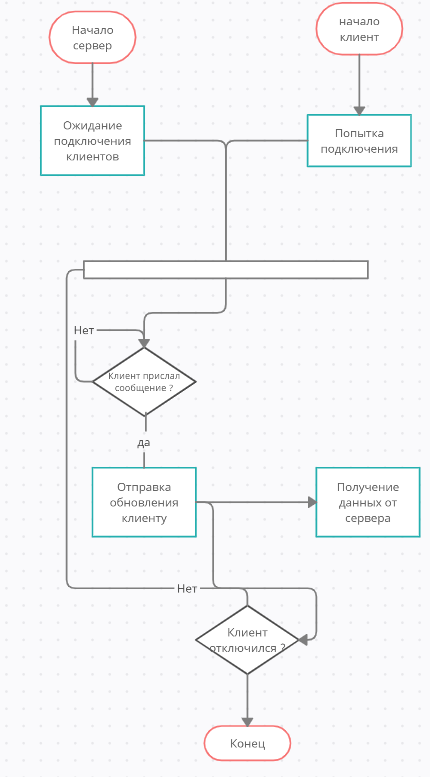
## Приложение Б.1 – MainframeClient.java

'use strict';  
  
var usernamePage = document.querySelector('#username-page');  
var chatPage = document.querySelector('#chat-page');  
var usernameForm = document.querySelector('#usernameForm');  
var messageForm = document.querySelector('#messageForm');  
var messageInput = document.querySelector('#message');  
var messageArea = document.querySelector('#messageArea');  
var connectingElement = document.querySelector('.connecting');  
  
var stompClient = null;  
var username = null;  
  
var colors = [  
 '#2196F3', '#32c787', '#00BCD4', '#ff5652',  
 '#ffc107', '#ff85af', '#FF9800', '#39bbb0'  
];  
  
function connect(event) {  
 username = document.querySelector('#name').value.trim();  
  
 if(username) {  
 usernamePage.classList.add('hidden');  
 chatPage.classList.remove('hidden');  
  
 var socket = new SockJS('/ws');  
 stompClient = Stomp.over(socket);  
  
 stompClient.connect({}, onConnected, onError);  
 }  
 event.preventDefault();  
}  
  
  
function onConnected() {  
 // Subscribe to the Public Topic  
 stompClient.subscribe('/topic/public', onMessageReceived);  
  
 // Tell your username to the server  
 stompClient.send("/app/chat.addUser",  
 {},  
 JSON.stringify({sender: username, type: 'JOIN'})  
 )  
  
 connectingElement.classList.add('hidden');  
}  
  
  
function onError(error) {  
 connectingElement.textContent = 'Could not connect to WebSocket server. Please refresh this page to try again!';  
 connectingElement.style.color = 'red';  
}  
  
  
function sendMessage(event) {  
 var messageContent = messageInput.value.trim();  
  
 if(messageContent && stompClient) {  
 var chatMessage = {  
 sender: username,  
 content: messageInput.value,  
 type: 'CHAT'  
 };  
  
 stompClient.send("/app/chat.sendMessage", {}, JSON.stringify(chatMessage));  
 messageInput.value = '';  
 }  
 event.preventDefault();  
}  
  
  
function onMessageReceived(payload) {  
 var message = JSON.parse(payload.body);  
  
 var messageElement = document.createElement('li');  
  
 if(message.type === 'JOIN') {  
 messageElement.classList.add('event-message');  
 message.content = message.sender + ' joined!';  
 } else if (message.type === 'LEAVE') {  
 messageElement.classList.add('event-message');  
 message.content = message.sender + ' left!';  
 } else {  
 messageElement.classList.add('chat-message');  
  
 var avatarElement = document.createElement('i');  
 var avatarText = document.createTextNode(message.sender[0]);  
 avatarElement.appendChild(avatarText);  
 avatarElement.style['background-color'] = getAvatarColor(message.sender);  
  
 messageElement.appendChild(avatarElement);  
  
 var usernameElement = document.createElement('span');  
 var usernameText = document.createTextNode(message.sender);  
 usernameElement.appendChild(usernameText);  
 messageElement.appendChild(usernameElement);  
 }  
  
 var textElement = document.createElement('p');  
 var messageText = document.createTextNode(message.content);  
 textElement.appendChild(messageText);  
  
 messageElement.appendChild(textElement);  
  
 messageArea.appendChild(messageElement);  
 messageArea.scrollTop = messageArea.scrollHeight;  
}  
  
  
function getAvatarColor(messageSender) {  
 var hash = 0;  
 for (var i = 0; i < messageSender.length; i++) {  
 hash = 31 \* hash + messageSender.charCodeAt(i);  
 }  
  
 var index = Math.abs(hash % colors.length);  
 return colors[index];  
}  
  
usernameForm.addEventListener('submit', connect, true)  
messageForm.addEventListener('submit', sendMessage, true)

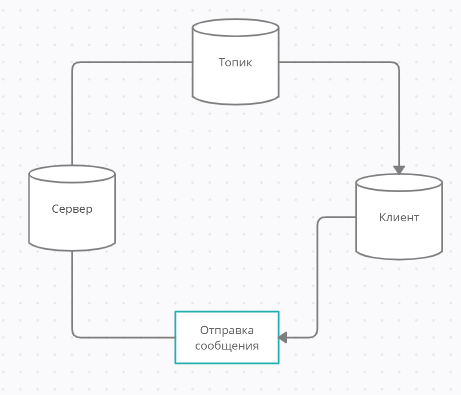
# Приложение В. UML-диаграммы приложения



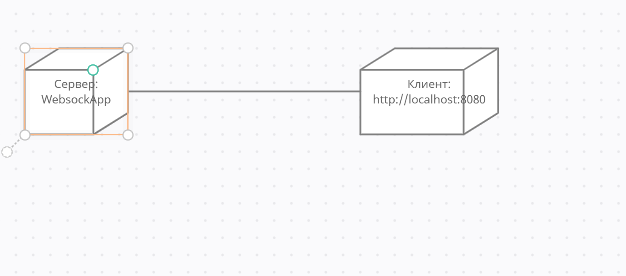
**UML-диаграммы взаимодействия**



**UML-диаграмма вариантов использования**



**UML-диаграмма развертывания**



**UML-диаграмма развертывания**

