**ИТМО Кафедра Информатики и прикладной математики**

Отчет по лабораторной работе №8

Программирование интернет-приложений

Вариант 6161

**Выполнил: студент группы P3217**

**Плюхин Дмитрий**

**Преподаватель: Гаврилов А. В.**

**2017 год**

1. **Задание к лабораторной работе**

Разработать приложение на базе JavaServer Faces Framework, которое осуществляет проверку попадания точки в заданную область на координатной плоскости.

Приложение должно включать в себя 2 facelets-шаблона - стартовую страницу и основную страницу приложения, а также набор управляемых бинов (managed beans), реализующих логику на стороне сервера.

Стартовая страница должна содержать следующие элементы:

"Шапку", содержащую ФИО студента, номер группы и номер варианта.

Интерактивные часы, показывающие текущие дату и время, обновляющиеся раз в 5 секунд.

Ссылку, позволяющую перейти на основную страницу приложения.

Основная страница приложения должна содержать следующие элементы:

Набор компонентов для задания координат точки и радиуса области в соответствии с вариантом задания. Может потребоваться использование дополнительных библиотек компонентов - ICEfaces (префикс "ace") и PrimeFaces (префикс "p"). Если компонент допускает ввод заведомо некорректных данных (таких, например, как буквы в координатах точки или отрицательный радиус), то приложение должно осуществлять их валидацию.

Динамически обновляемую картинку, изображающую область на координатной плоскости в соответствии с номером варианта и точки, координаты которых были заданы пользователем. Клик по картинке должен инициировать сценарий, осуществляющий определение координат новой точки и отправку их на сервер для проверки её попадания в область. Цвет точек должен зависить от факта попадания / непопадания в область. Смена радиуса также должна инициировать перерисовку картинки.

Таблицу со списком результатов предыдущих проверок.

Ссылку, позволяющую вернуться на стартовую страницу.

Дополнительные требования к приложению:

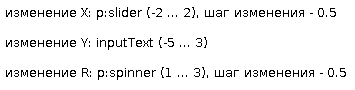
Все результаты проверки должны сохраняться в базе данных под управлением СУБД PostgreSQL.

Для доступа к БД необходимо использовать протокол JDBC без каких-либо дополнительных библиотек.

Для управления списком результатов должен использоваться Session-scoped Managed Bean.

Конфигурация управляемых бинов должна быть задана с помощью аннотаций.

Правила навигации между страницами приложения должны быть заданы в отдельном конфигурационном файле.



1. **Исходный код**

**//Файл App.xhtml**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core"

xmlns:p="http://primefaces.org/ui"

xmlns:a4j="http://richfaces.org/a4j"

xmlns:rich="http://richfaces.org/rich"

xmlns:ui="http://java.sun.com/jsf/facelets">

<h:head>

<title>Silhouette checker</title>

<link href="#{request.contextPath}/resources/css/style.css" rel="stylesheet" type="text/css"></link>

</h:head>

<h:body>

<h:outputScript library="primefaces" name="primefaces.js" />

<h:outputScript library="primefaces" name="jquery/jquery.js" target="head" />

<h1>Silhouette checker</h1>

<h:outputStylesheet name="styles.css" />

<h:form id="gui">

<h:graphicImage id="graph" value="resources/images/area.png">

<p:ajax event="click" update="graph,:manual:tabledata"/>

<h:panelGroup id="projects" styleClass="projects">

<ui:repeat value = "#{pontoBean.previous}" var = "c">

<h:panelGroup layout="block" styleClass="positivedot" style="margin-left:#{c.x\*80/pontoBean.actualR+101}px;margin-top:#{-c.y\*80/pontoBean.actualR+105}px;" rendered="#{c.inside}"></h:panelGroup>

<h:panelGroup layout="block" styleClass="negativedot" style="margin-left:#{c.x\*80/pontoBean.actualR+101}px;margin-top:#{-c.y\*80/pontoBean.actualR+105}px;" rendered="#{not c.inside}"></h:panelGroup>

</ui:repeat>

</h:panelGroup>

</h:graphicImage>

<h:inputText id="x" style="display:none" value="#{pontoBean.actualX}" />

<h:inputText id="y" style="display:none" value="#{pontoBean.actualY}" />

<h:commandButton style="display:none" id="gui\_submit" value="Submit" action="#{pontoBean.checkPonto}" />

</h:form>

<h:form id="manual">

<h:inputHidden id="txt2" value="#{pontoBean.actualX}" />

<p:panel>

<h:panelGrid columns="1">

<h:outputText id="outputX" value="X : "/>

<p:slider for="txt2" minValue="-2" maxValue="2" step="1" display="outputX" style="width: 200px" displayTemplate="X : {value}" />

<h:outputText value="Y : "/>

<h:inputText id="ordinate" value="#{pontoBean.actualY}">

<f:validator validatorId="ru.jsf.YValidator" />

</h:inputText>

<h:outputText value="R : "/>

<p:spinner label="radius" id="radius" value="#{pontoBean.actualR}" min="1" max="3" stepFactor="0.5">

<p:ajax event="change" listener="#{pontoBean.updateData()}" update=":gui:graph,tabledata"/>

</p:spinner>

</h:panelGrid>

<h:message for="ordinate" style="color:red" />

<h:commandButton value="Submit" action="#{pontoBean.checkPonto}" />

<h:commandButton styleClass="bigBlueButton" id="back" value="BACK" action="showStart"/>

</p:panel>

<h:dataTable id="tabledata" value = "#{pontoBean.previous}" var = "c">

<h:column>

<f:facet name = "header">X</f:facet>

#{c.x}

</h:column>

<h:column>

<f:facet name = "header">Y</f:facet>

#{c.y}

</h:column>

<h:column>

<f:facet name = "header">R</f:facet>

#{c.r}

</h:column>

<h:column>

<f:facet name = "header">Inside</f:facet>

#{c.inside}

</h:column>

</h:dataTable>

</h:form>

<h:outputScript>

var deletingCounter = 0;

console.log($("input[id$=':radius\_input']"));

$("img[id$=':graph']").bind("DOMSubtreeModified",function(){

if ((deletingCounter > 0) &amp;&amp; (($("div.negativedot").length + $("div.negativedot").length) > 0)){

deletingCounter--;

$("div.negativedot").remove();

$("div.positivedot").remove();

$("span.projects").each(function(){

if( $.trim($(this).text()) == "" ){

$(this).remove();

}

});

}

});

$("input[id$=':radius\_input']").on("change", function(event){

console.log("--");

if (($("div.negativedot").length + $("div.negativedot").length) > 0){

$("div.negativedot").remove();

$("div.positivedot").remove();

$("span.projects").each(function(){

if( $.trim($(this).text()) == "" ){

$(this).remove();

}

});

} else {

deletingCounter++;

}

});

$("img[id$=':graph']").on("mousedown", function(event) {

console.log("ok");

$form = $("form#gui");

$formtwo = $("form#manual");

console.log($form);

console.log((event.pageX - event.target.offsetLeft - 103)\*($formtwo.find("input[id$=':radius\_input']").val()/80));

$form.find("input[id$=':x']").val((event.pageX - event.target.offsetLeft - 103)\*($formtwo.find("input[id$=':radius\_input']").val()/80));

$form.find("input[id$=':y']").val((-(event.pageY - event.target.offsetTop) + 107)\*($formtwo.find("input[id$=':radius\_input']").val()/80));

$("input[id$=':gui\_submit']").click();

});

</h:outputScript>

</h:body>

</html>

**//Файл PontoBean.java**

package ru.jsf;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.faces.bean.ManagedBean;

import javax.faces.bean.SessionScoped;

import java.util.Date;

import javax.faces.application.FacesMessage;

import javax.faces.context.FacesContext;

import org.primefaces.event.SlideEndEvent;

import silhouette.\*;

@ManagedBean

@SessionScoped

public class PontoBean{

private double actualX;

private double actualY;

private double actualR;

private GeneralSilhouette slh;

private Connection con;

public PontoBean(){

setActualR("1.0");

}

public String getActualX(){

return String.valueOf(actualX);

}

public void setActualX(String actualX){

this.actualX = Double.parseDouble(actualX);

}

public String getActualY(){

return String.valueOf(actualY);

}

public void setActualY(String actualY){

this.actualY = Double.parseDouble(actualY);

}

public String getActualR(){

return String.valueOf(actualR);

}

public void setActualR(String actualR){

this.actualR = Double.parseDouble(actualR);

slh = new GeneralSilhouette(this.actualR);

}

public void truncateTable() throws SQLException, ClassNotFoundException{

String query = "TRUNCATE TABLE pontos";

Connection con = getConnection();

PreparedStatement pst = con.prepareStatement(query);

pst.execute();

//con.close();

}

public List<PontoInfo> getPrevious() throws SQLException, ClassNotFoundException{

String query = "SELECT \* FROM pontos";

Connection con = getConnection();

PreparedStatement pst = con.prepareStatement(query);

ResultSet rs = pst.executeQuery();

List<PontoInfo> pinfos = new ArrayList<PontoInfo>();

while(rs.next()) {

PontoInfo pinfo = new PontoInfo();

pinfo.setX(rs.getDouble(1));

pinfo.setY(rs.getDouble(2));

pinfo.setR(rs.getDouble(3));

pinfo.setInside(rs.getBoolean(4));

pinfos.add(pinfo);

}

//con.close();

return pinfos;

}

public String updateData() throws SQLException, ClassNotFoundException{

List<PontoInfo> pinfos = getPrevious();

truncateTable();

for(PontoInfo pinfo : pinfos){

actualX = pinfo.getX();

actualY = pinfo.getY();

checkPonto();

}

return "successful";

}

public String checkPonto() throws SQLException, ClassNotFoundException{

boolean result = slh.checkPonto(new Ponto(actualX, actualY));

String query = "INSERT INTO pontos (x, y, r, inside) values('"+getActualX()+"', '"+getActualY()+"', '"+getActualR()+"', '"+result+"')";

Connection con = getConnection();

PreparedStatement pst = con.prepareStatement(query);

pst.execute();

//con.close();

if(result) return "successful";

return "fail";

}

public Connection getConnection() throws SQLException, ClassNotFoundException{

if ((con == null) || con.isClosed()){

Class.forName("org.postgresql.Driver");

String url = "jdbc:postgresql://localhost/test";

String user = "s207602";

String password = "gyp849";

con = DriverManager.getConnection(url, user, password);

}

return con;

}

}

**//Файл main.xhtml**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:p="http://primefaces.prime.com.tr/ui"

xmlns:a4j="http://richfaces.org/a4j"

xmlns:rich="http://richfaces.org/rich">

<h:head>

<title>Welcome</title>

<link href="#{request.contextPath}/resources/css/style.css" rel="stylesheet" type="text/css"></link>

</h:head>

<h:body>

<h:outputScript library="primefaces" name="primefaces.js" />

<h1>Выполнил студент группы P3217 Плюхин Дмитрий Алексеевич вариант 6161</h1>

<h:outputStylesheet name="styles.css" />

<h:form>

<a4j:poll id="poll" interval="5000" render="poll,time" />

<p:panel>

<h:panelGrid columns="2">

<h:outputText styleClass="timer" value="Server date and time : "/>

<h:outputText styleClass="timer" id="time" value="#{clockBean.serverTime}"/>

</h:panelGrid>

<h:commandButton styleClass="bigBlueButton" id="ok" value="OK" action="showApp"/>

</p:panel>

</h:form>

</h:body>

</html>

**//Файл YValidator.java**

package ru.jsf;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import javax.faces.application.FacesMessage;

import javax.faces.component.UIComponent;

import javax.faces.context.FacesContext;

import javax.faces.validator.FacesValidator;

import javax.faces.validator.Validator;

import javax.faces.validator.ValidatorException;

@FacesValidator("ru.jsf.YValidator")

public class YValidator implements Validator{

private void throwError(){

FacesMessage msg = new FacesMessage("Y validation failed.","Invalid Y value.");

msg.setSeverity(FacesMessage.SEVERITY\_ERROR);

throw new ValidatorException(msg);

}

@Override

public void validate(FacesContext context, UIComponent component,Object value) throws ValidatorException {

double validated = 0;

try{

validated = Double.parseDouble(value.toString());

} catch (NumberFormatException e){

throwError();

}

if ((validated < -5) || (validated > 3)) throwError();

}

}

**//Файл ClockBean.java**

package ru.jsf;

import javax.faces.bean.ManagedBean;

import javax.faces.bean.SessionScoped;

import java.util.Date;

@ManagedBean

@SessionScoped

public class ClockBean{

private String serverTime = "";

public void updateServerTime(){

Date date = new Date();

serverTime = date.toString();

}

public String getServerTime(){

updateServerTime();

return serverTime;

}

public void setServerTime(String time){

Date date = new Date();

serverTime = date.toString();

}

}