Санкт-Петербургский национальный исследовательский университет

информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

**Дисциплина «Программирование интернет-приложений»**

**Лабораторная работа №2**

**Вариант 713**

**Выполнили:**

Ореховский Антон

Чемыртан Андрей

Группа P3217

**Преподаватель:**

Николаев Владимир Вячеславович

Санкт-Петербург

2017

Разработать веб-приложение на базе сервлетов и JSP, определяющее попадание точки на координатной плоскости в заданную область.

Приложение должно быть реализовано в соответствии с шаблоном MVC и состоять из следующих элементов:

* **ControllerServlet**, определяющий тип запроса, и, в зависимости от того, содержит ли запрос информацию о координатах точки и радиусе, делегирующий его обработку одному из перечисленных ниже компонентов. Все запросы внутри приложения должны передаваться этому сервлету (по методу GET), остальные сервлеты с веб-страниц напрямую вызываться не должны.
* **AreaCheckServlet**, осуществляющий проверку попадания точки в область на координатной плоскости и формирующий HTML-страницу с результатами проверки. Должен обрабатывать все запросы, содержащие сведения о координатах точки и радиусе области.
* **Страница JSP**, формирующая HTML-страницу с веб-формой. Должна обрабатывать все запросы, не содержащие сведений о координатах точки и радиусе области.

**Разработанная страница JSP должна содержать:**

1. "Шапку", содержащую ФИО студента, номер группы и номер варианта.
2. Форму, отправляющую данные на сервер.
3. Набор полей для задания координат точки и радиуса области в соответствии с вариантом задания.
4. Сценарий на языке JavaScript, осуществляющий валидацию значений, вводимых пользователем в поля формы.
5. Интерактивный элемент, содержащий изображение области на координатной плоскости (в соответствии с вариантом задания) и реализующий следующую функциональность:
   * Если радиус области установлен, клик курсором мыши по изображению должен обрабатываться JavaScript-функцией, определяющей координаты точки, по которой кликнул пользователь и отправляющей полученные координаты на сервер для проверки факта попадания.
   * В противном случае, после клика по картинке должно выводиться сообщение о невозможности определения координат точки.
   * После проверки факта попадания точки в область изображение должно быть обновлено с учётом результатов этой проверки (т.е., на нём должна появиться новая точка).
6. Таблицу с результатами предыдущих проверок. Список результатов должен браться из контекста приложения, HTTP-сессии.

**Страница, возвращаемая AreaCheckServlet, должна содержать:**

1. Таблицу, содержащую полученные параметры.
2. Результат вычислений - факт попадания или непопадания точки в область.
3. Ссылку на страницу с веб-формой для формирования нового запроса.

Разработанное веб-приложение необходимо развернуть на сервере GlassFish.

**ControllerServlet (Controller)**

public class ControllerServlet extends HttpServlet

{

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException

{

RequestDispatcher dispatcher;

String rString = request.getParameter("r"),

xString = request.getParameter("x"),

yString = request.getParameter("y");

float r;

try

{

assert (rString != null && xString != null && yString != null);

r = Float.parseFloat(rString);

Float.parseFloat(xString);

Float.parseFloat(yString);

assert (r >= 0);

}

catch (Exception exception)

{

dispatcher = request.getRequestDispatcher("/index.jsp");

dispatcher.forward(request, response);

return;

}

dispatcher = request.getRequestDispatcher("/check");

dispatcher.forward(request, response);

}

}

**AreaCheckServlet (Model)**

public class AreaCheckServlet extends HttpServlet

{

private boolean checkArea(float r, Point p)

throws IOException

{

if (Math.pow(p.y, 2) + Math.pow(p.x, 2) <= Math.pow(r, 2) && p.x >= 0 && p.y >= 0 ||

p.y >= 0.5 \* p.x - (double)(r / 2) && p.x >= 0 && p.y <= 0 ||

Math.abs(p.y) <= r && Math.abs(p.x) <=((double) r / 2) && p.x <= 0 && p.y <=0)

return true;

else

return false;

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException

{

String rStr = request.getParameter("r\_field"),

xStr = request.getParameter("x"),

yStr = request.getParameter("y");

float r = Float.NaN, x = Float.NaN, y = Float.NaN;

boolean error = false, result = false;

try {

r = Float.parseFloat(rStr.replace(',', '.'));

x = Float.parseFloat(xStr.replace(',', '.'));

y = Float.parseFloat(yStr.replace(',', '.'));

Point p = new Point(x, y);

result = checkArea(r, p);

}

catch (Exception e) { error = true; }

if (r < 0 || Math.abs(x) > 5 || Math.abs(y) > 5) error = true;

PrintWriter out = response.getWriter();

//выводим страницу

response.setContentType("text/html");

out.print("<!DOCTYPE html>" +

"<html lang=\"en\">" +

"<style type=\"text/css\">" +

" div {" +

" margin: 5% auto 5%" +

" }" +

" table {" +

" border: 1px solid #0c181f;" +

" border-collapse: collapse;" +

" margin: inherit;" +

" }" +

" th, td {" +

" border: 1px solid #0c181f;" +

" border-collapse: collapse;" +

" text-align: center;" +

" padding: 15px;" +

" }" +

" button {" +

" background-color: #0c181f;" +

" width: 150px;" +

" height: 40px;" +

" color: white;" +

" font-size: 16;" +

" font-family: \"Fantasy\";" +

" font-weight: bold;" +

" border: none;" +

" border-radius: 15px;" +

"}" +

"</style>" +

"<head>" +

" <meta charset=\"UTF-8\">" +

" <title>Results</title>" +

"</head>" +

"<body>" +

" <div> " +

" <table border=\"1\">" +

" <tr>" +

" <th>Radius</th>" +

" <th>X</th>" +

" <th>Y</th>" +

" <th>Included?</th>" +

" </tr>" +

" <tr>" +

" <td>" + (!Float.isNaN(r) ? r : rStr) + "</td>" +

" <td>" + (!Float.isNaN(x) ? x : xStr) + "</td>" +

" <td>" + (!Float.isNaN(y) ? y : yStr) + "</td>" +

" <td>" + (error ? "Error!" : result ? "Yes" : "No") + "</td>" +

" </tr>" +

" </table>" +

" </div>" +

" <div style=\"text-align: center\">" +

" <button onclick=\"location.href='./';\">Return</button>" +

" </div>" +

"</body>" +

"</html>");

out.close();

if (error) return;

//сохраняем данные в контекст

ServletContext context = getServletContext();

ArrayList<Float> rList = (ArrayList)context.getAttribute("r");

ArrayList<Float> xList = (ArrayList)context.getAttribute("x");

ArrayList<Float> yList = (ArrayList)context.getAttribute("y");

ArrayList<Boolean> resList = (ArrayList)context.getAttribute("result");

if(rList == null || xList == null || yList == null || resList == null)

{

rList = new ArrayList<Float>(10);

xList = new ArrayList<Float>(10);

yList = new ArrayList<Float>(10);

resList = new ArrayList<Boolean>(10);

}

rList.add(r);

xList.add(x);

yList.add(y);

resList.add(result);

context.setAttribute("r", rList);

context.setAttribute("x", xList);

context.setAttribute("y", yList);

context.setAttribute("result", resList);

}

}

**JSP-страница**

<%@ page import="java.util.ArrayList" %>

<script src="https://code.jquery.com/jquery-2.2.3.min.js"></script>

<html>

<head>

<meta charset="UTF-8">

<link href="style.css" rel="stylesheet">

<title>Laboratory work #2</title>

</head>

<body>

<header id="index\_header">

<div class="logo"><img src="itmo\_header.png" width="120px" height="60px" ></div>

<div class="lab">Lab 2</div>

<div class="authors">Chemyrtan Andrey, Orekhovskiy Anton P3217</div>

</header>

<img src="axis.png" id="axis" style="visibility: hidden; width: 0; height: 0">

<div class="outer" id="main\_block">

<div class="inner" id="canvas\_block">

<canvas id="canvas" width="500px" height="500px"> </canvas>

<script>

var canvas = document.getElementById("canvas"),

context = canvas.getContext("2d");

context.font = "24px Arial";

context.textAlign = "center";

const center = 250, cellSize = 50, dotRadius = 2;

function drawArea(r) {

r = r \* cellSize;

context.clearRect(0, 0, 600, 600);

var radiusDefined = typeof(r) !== "undefined";

if (radiusDefined) {

context.fillStyle = "rgb(0, 0, 255)";

context.fillRect(250 - r / 2, 250, r / 2, r);

context.beginPath();

context.arc(250, 250, r, 0, 3/2 \* Math.PI, true);

context.lineTo(250, 250 + r/2);

context.fill();

context.closePath();

}

drawAxis();

drawPoints(r);

}

function drawAxis() {

var axis = document.getElementById("axis");

context.drawImage(axis,0,0, 500, 500);

}

function areaCheck(x,y,r) {

if (Math.pow(y, 2) + Math.pow(x, 2) <= Math.pow(r, 2) && x >= 0 && y >= 0 ||

y >= 0.5 \* x - (r / 2) && x >= 0 && y <= 0 ||

Math.abs(y) <= r && Math.abs(x) <= r / 2 && x <= 0 && y <=0)

return true;

else

return false;

}

function drawPoints(r) {

$('table > tbody > tr').each(function(index, element) {

var x = parseFloat(element.cells[1].innerHTML),

y = parseFloat(element.cells[2].innerHTML);

var result = areaCheck(x, y, r / cellSize);

context.fillStyle = (result ? "Green" : "Red");

context.beginPath();

context.arc(center + x \* cellSize ^ 0, center - y \* cellSize ^ 0, dotRadius, 0, 2 \* Math.PI);

context.fill();

context.closePath();

});

}

function updateCanvas() {

var r = null;

var input = document.getElementsByName("r\_field");

for (var i = 0; i < input.length; i++){

if (input[i].checked){

r = input[i].value;

break;

}

}

var radiusDefined = r >= 1 && r <= 5 && r!=null;

drawArea(radiusDefined ? r : undefined);

}

function getMousePos(canvas, e) {

var rect = canvas.getBoundingClientRect();

return {

x: e.clientX - rect.left,

y: e.clientY - rect.top

};

}

canvas.addEventListener("click", canvasClickEvent, false);

function canvasClickEvent(e) {

var r = null;

var input = document.getElementsByName("r\_field");

for (var i = 0; i < input.length; i++){

if (input[i].checked){

r = input[i].value;

break;

}

}

if (r!=null && r >= 1 && r <= 5)

{

var coordinates = getMousePos(canvas, e);

document.forms["form"]["x\_field"][0].value = (coordinates.x - center) / cellSize;

document.forms["form"]["y\_field"].value = (center - coordinates.y) / cellSize;

document.forms["form"].submit();

}

else alert("Unable to get coordinates")

}

</script>

</div>

<div class="inner" style="padding-left: 150px">

<div id="validation\_block">

<form name="form" action="./check" onsubmit="return validate()" method="get">

<p id="info\_field">Input parameters:</p>

Input radius:

<input type="radio" name="r\_field" value="1" onclick="updateCanvas();">1

<input type="radio" name="r\_field" value="2" onclick="updateCanvas();">2

<input type="radio" name="r\_field" value="3" onclick="updateCanvas();">3

<input type="radio" name="r\_field" value="4" onclick="updateCanvas();">4

<input type="radio" name="r\_field" value="5" onclick="updateCanvas();">5

<br><br>

Input coordinate X:

<select name="x" id="x\_field">

<option value="-3">-3</option>

<option value="-2">-2</option>

<option value="-1">-1</option>

<option value="0">0</option>

<option value="1">1</option>

<option value="2">2</option>

<option value="3">3</option>

<option value="4">4</option>

<option value="5">5</option>

</select>

<br><br>

Input coordinate Y:

<input type="text" id="y\_field" name="y">

<br><br>

<input type="submit" value="Send" class="button">

</form>

<script>

function setInfoText(description) {

document.getElementById("info\_field").innerHTML = description;

return false;

}

function validate() {

const int\_regex = /^0|-?[1-9]\d\*$/,

real\_regex = /^0|-?(?:(?:[1-9]\d\*(?:[,.]\d+)?)|0[,.]\d+)$/;

if (!int\_regex.test(document.forms["form"]["r\_field"].value))

return setInfoText("Incorrect input in radius field");

if (!real\_regex.test(document.forms["form"]["x\_field"].value))

return setInfoText("Incorrect input in X field");

if (!real\_regex.test(document.forms["form"]["y\_field"].value))

return setInfoText("Incorrect input in Y field");

var r = parseInt(document.forms["form"]["r\_field"].value);

if (r < 1 || r > 5) return setInfoText("Radius must be an integer between 1 and 5");

var x = parseFloat(document.forms["form"]["x\_field"].value);

if (x < -5 || x > 5) return setInfoText("X must be between -5 and 5");

var y = parseFloat(document.forms["form"]["y\_field"].value);

if (y < -5 || y > 5) return setInfoText("Y must be between -5 and 5");

}

</script>

</div>

</div>

</div>

<div>

<table id="table">

<thead>

<tr>

<th>Radius</th>

<th>X</th>

<th>Y</th>

<th>Included?</th>

</tr>

</thead>

<tbody>

<%

ServletContext context = session.getServletContext();

try {

ArrayList<Float> r = (ArrayList<Float>) context.getAttribute("r"),

x = (ArrayList) context.getAttribute("x"),

y = (ArrayList) context.getAttribute("y");

ArrayList<Boolean> result = (ArrayList) context.getAttribute("result");

int count = r.size();

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

String resultStr = result.get(i) ? "Yes" : "No";

out.println("<tr>" +

" <td>" + r.get(i) + "</td>" +

" <td>" + x.get(i) + "</td>" +

" <td>" + y.get(i) + "</td>" +

" <td class=\"" + resultStr + "Row\">" + resultStr + "</td>" +

"</tr>");

}

}

catch (Exception e) {}

%>

</tbody>

</table>

</div>

</body>

</html>

**Вывод**

В ходе выполнения данной работы мы научились разрабатывать веб-приложения на основе сервлетов, соответствующие модели MVC, а также обрабатывать события с помощью Javascript.