Министерство образования и науки Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №6

по курсу "Разработка web-приложений"  
на тему "Реализация REST API"

Выполнили:

Студенты группы 16ВВ2

Воронков Е.В.

Духанов И.В.

Проверили:

Дубравин А.В.  
Карамышева Н.С.

Пенза 2020

**Цель работы**

Изучить основы обработки запросов клиента, реализовав в ней поддержку REST API. Вариант - 2.

**Задание:**

Необходимо доработать лабораторную работу №5 реализовав в ней поддержку REST API и реализовать два обработчика:

1. Получает запрос с названием животного и дополнительными параметрами и подбирает по этому запросу подходящее изображение из числа имеющихся;

2. По имени и параметрам изображения возвращает картинку.

Параметры должны передаваться на сервер в виде POST запроса, в теле которого должен находиться JSON.

Ответ от сервера так же должен возвращаться в формате JSON. Необходимо расширить набор параметров изображения животного, добавив в него не обязательные значения цвета и размера ожидаемого изображения (количество пикселей по горизонтали и по вертикали).

В ответ на запрос сервер должен сформировать ответ, в котором перечислить переданные параметры, а недостающие заполнить случайными значениями (случайные значения должны выбираться из разумного диапазона), а также должен добавить поле с именем файла, соответствующего заданным параметрам.

**Выполнение:**

Создали две формы запроса, для поиска по параметрам содержимого изображения и для поиска по параметрам самого изображения. Заполнив все обязательные поля запроса и нажав кнопку Submit на сервер отправляется POST-запрос, в теле которого содержится JSON со всеми введенными параметрами. Если обязательные поля были заполнены не верно, сервер возвращает JSON в котором содержится свойство error, в котором описана ошибка. Если обязательные поля были заполнены верно, на сервере все пустые дополнительные поля заполняются случайным допустимым значением и производится поиск соответствующего изображения в JSON ответа добавляется свойство imgURL содержащее URL искомого изображения. Если изображение не было найдено в JSON ответа добавляется свойство error сообщающее об ошибке поиска.

Листинг файла index.js:

const server = require("./server");

const router = require("./router");

const requestHandlers = require("./requestHandlers");

var handle = {}

handle["/"] = requestHandlers.start;

handle["/start"] = requestHandlers.start;

handle["/favicon.ico"] = requestHandlers.favicon;

handle["/byContent"] = requestHandlers.byContent;

handle["/byImage"] = requestHandlers.byImage;

server.start(router.route, handle);

Листинг файла router.js:

const fs = require("fs");

function route(handle, pathname, response, postData) {

console.log("About to route a request for " + pathname);

if (pathname.endsWith('.jpg')){

response.write(fs.readFileSync(pathname.slice(1)));

response.end();

return;

}

if (typeof handle[pathname] === 'function') {

handle[pathname](response, postData);

}

else {

console.log("No request handler found for " + pathname);

response.writeHead(404, {"Content-Type": "text/plain"});

response.write("404 Not found");

response.end();

}

}

exports.route = route

Листинг файла requestHandler.js:

const fs = require("fs");

function start(response, postData) {

console.log("Request handler 'start' was called.");

const page = fs.readFileSync('index.html');

response.writeHead(200, {"Content-Type": "text/html"});

response.write(page);

response.end();

}

function favicon(response) {

console.log("Request handler 'favicon' was called.");

response.writeHead(404);

response.end();

}

function beaver(response, idPic)

{

response.write(fs.readFileSync('images/beaver/'+idPic+'.jpg'));

}

function bison(response, idPic)

{

response.write(fs.readFileSync('images/bison/'+idPic+'.jpg'));

}

function hippo(response, idPic)

{

response.write(fs.readFileSync('images/hippo/'+idPic+'.jpg'));

}

function formatError(response)

{

response.write(JSON.stringify({error: "Неверный формат запроса"}));

response.end();

}

function byContent(response, postData)

{

var images = {

WhiteBeaverS: {animal: "beaver", colour: "white", grouping: "single", url: "images/WhiteBeaverS.jpg"},

WhiteBeaverM: {animal: "beaver", colour: "white", grouping: "multiple", url: "images/WhiteBeaverM.jpg"},

BlackBeaverS: {animal: "beaver", colour: "black", grouping: "single", url: "images/BlackBeaverS.jpg"},

BlackBeaverM: {animal: "beaver", colour: "black", grouping: "multiple", url: "images/BlackBeaverM.jpg"},

PurpleBeaverS: {animal: "beaver", colour: "purple", grouping: "single", url: "images/PurpleBeaverS.jpg"},

WhiteBisonS: {animal: "bison", colour: "white", grouping: "single", url: "images/WhiteBisonS.jpg"},

BlackBisonS: {animal: "bison", colour: "black", grouping: "single", url: "images/BlackBisonS.jpg"},

BlackBisonM: {animal: "bison", colour: "black", grouping: "multiple", url: "images/BlackBisonM.jpg"},

PurpleBisonS: {animal: "bison", colour: "purple", grouping: "single", url: "images/PurpleBisonS.jpg"},

PurpleBisonM: {animal: "bison", colour: "purple", grouping: "multiple", url: "images/PurpleBisonM.jpg"},

WhiteHippoS: {animal: "hippo", colour: "white", grouping: "single", url: "images/WhiteHippoS.jpg"},

WhiteHippoM: {animal: "hippo", colour: "white", grouping: "multiple", url: "images/WhiteHippoM.jpg"},

BlackHippoS: {animal: "hippo", colour: "black", grouping: "single", url: "images/BlackHippoS.jpg"},

BlackHippoM: {animal: "hippo", colour: "black", grouping: "multiple", url: "images/BlackHippoM.jpg"},

PurpleHippoS: {animal: "hippo", colour: "purple", grouping: "single", url: "images/PurpleHippoS.jpg"},

PurpleHippoM: {animal: "hippo", colour: "purple", grouping: "multiple", url: "images/PurpleHippoM.jpg"},

}

var properties = JSON.parse(postData);

console.log(properties);

if (properties.animal !== "beaver" && properties.animal !== "bison" && properties.animal !== "hippo"){

formatError(response);

return;

}

if(JSON.parse(postData).colour === ''){

var id = Math.floor(Math.random() \* Math.floor(3));

switch (id) {

case 0:

properties.colour = "white";

break;

case 1:

properties.colour = "black";

break;

case 2:

properties.colour = "purple";

break;

default:

console.log("RandomColorError");

}

}

if(JSON.parse(postData).grouping === ''){

var id = Math.floor(Math.random() \* Math.floor(2));

switch (id) {

case 0:

properties.grouping = "single";

break;

case 1:

properties.grouping = "multiple";

break;

default:

console.log("RandomGroupingError");

}

}

console.log(properties);

for (var image in images){

if(properties.animal === images[image].animal &&

properties.colour === images[image].colour &&

properties.grouping === images[image].grouping){

response.writeHead(200);

console.log(JSON.parse(postData));

response.write(JSON.stringify({ animal: properties.animal, colour: properties.colour, grouping: properties.grouping, imgURL: images[image].url}));

response.end();

return;

}

}

response.write(JSON.stringify({ animal: properties.animal, colour: properties.colour, grouping: properties.grouping, error: "Изображение не найдено"}));

response.end();

}

function byImage(response, postData)

{

var images = {

WhiteBeaverS: {filename: "WhiteBeaverS.jpg", width: 1280, height: 720},

WhiteBeaverM: {filename: "WhiteBeaverM.jpg", width: 800, height: 533},

BlackBeaverS: {filename: "BlackBeaverS.jpg", width: 800, height: 532},

BlackBeaverM: {filename: "BlackBeaverM.jpg", width: 1952, height: 1281},

PurpleBeaverS: {filename: "PurpleBeaverS.jpg", width: 900, height: 900},

WhiteBisonS: {filename: "WhiteBisonS.jpg", width: 1600, height: 1599},

BlackBisonS: {filename: "BlackBisonS.jpg", width: 3888, height: 2592},

BlackBisonM: {filename: "BlackBisonM.jpg", width: 1140, height: 712},

PurpleBisonS: {filename: "PurpleBisonS.jpg", width: 508, height: 495},

PurpleBisonM: {filename: "PurpleBisonM.jpg", width: 533, height: 800},

WhiteHippoS: {filename: "WhiteHippoS.jpg", width: 736, height: 736},

WhiteHippoM: {filename: "WhiteHippoM.jpg", width: 2433, height: 1414},

BlackHippoS: {filename: "BlackHippoS.jpg", width: 2000, height: 1648},

BlackHippoM: {filename: "BlackHippoM.jpg", width: 1200, height: 675},

PurpleHippoS: {filename: "PurpleHippoS.jpg", width: 800, height: 800},

PurpleHippoM: {filename: "PurpleHippoM.jpg", width: 238, height: 250}

}

var properties = JSON.parse(postData);

for (var image in images){

if(properties.filename === images[image].filename &&

properties.width === images[image].width &&

properties.height === images[image].height){

response.writeHead(200);

console.log(JSON.parse(postData));

response.write(JSON.stringify({ filename: properties.filename, width: properties.width, height: properties.height, imgURL: "images/" + properties.filename}));

response.end();

return;

}

}

response.write(JSON.stringify({ animal: properties.animal, colour: properties.colour, grouping: properties.grouping, error: "Изображение не найдено"}));

response.end();

}

exports.start = start;

exports.favicon = favicon;

exports.byContent = byContent;

exports.byImage = byImage;

Листинг файла server.js:

var http = require("http");

var url = require("url");

function start(route, handle) {

function onRequest(request, response) {

var postData = "";

var pathname = url.parse(request.url).pathname;

console.log("Request for " + pathname + " received.");

request.setEncoding("utf8");

request.addListener("data", function(postDataChunk) {

postData += postDataChunk;

console.log("Received POST data chunk '"+

postDataChunk + "'.");

});

request.addListener("end", function() {

route(handle, pathname, response, postData);

});

}

http.createServer(onRequest).listen(8888);

console.log("Server has started.");

}

exports.start = start;

Листинг файла index.html:

<html>

<head>

<title>WowReturns</title>

<meta http-equiv="Content-Type" content="text/html;'charset=UTF-8" />

</head>

<script>

document.addEventListener('click', () =>

{

if (!event.target.matches('.sndCnt')) return;

event.preventDefault();

let url = new URL('http://localhost:8888/byContent');

let xhr = new XMLHttpRequest();

xhr.open('POST', url);

var grp = '';

var message = JSON.stringify({

animal: document.getElementById("animal").value,

colour: document.getElementById("colour").value,

grouping: document.getElementsByName("grouping")[0].checked ?

document.getElementsByName("grouping")[0].value :

document.getElementsByName("grouping")[1].checked ?

document.getElementsByName("grouping")[1].value :

""

});

xhr.send(message);

xhr.onerror = function() {

alert('Ошибка соединения');

};

xhr.onload = function() {

var animal = JSON.parse(xhr.response).animal;

var colour = JSON.parse(xhr.response).colour;

var grouping = JSON.parse(xhr.response).grouping;

console.log(xhr.response);

if(!JSON.parse(xhr.response).error){

document.getElementById("resp").innerHTML = "<strong>Ответ на запрос:</strong><br /><br />" +

"Название животного: <strong>" + animal + "</strong><br />" +

"Цвет: <strong>" + colour + "</strong><br />" +

"Количество животных: <strong>" + grouping + "</strong><br />";

document.getElementById("myimage").innerHTML = '<img src="' + JSON.parse(xhr.response).imgURL + '" alt="" width="400">';

}

else{

document.getElementById("resp").innerHTML = "<strong>" + JSON.parse(xhr.response).error + "</strong>";

document.getElementById("myimage").innerHTML = "";

}

};

});

</script>

<script>

document.addEventListener('click', () =>

{

if (!event.target.matches('.sndImg')) return;

event.preventDefault();

let url = new URL('http://localhost:8888/byImage');

let xhr = new XMLHttpRequest();

xhr.open('POST', url);

var message = JSON.stringify({

filename: document.getElementById("filename").value,

width: parseInt(document.getElementById("width").value),

height: parseInt(document.getElementById("height").value)

});

xhr.send(message);

xhr.onerror = function() {

alert('Ошибка соединения');

};

xhr.onload = function() {

var filename = JSON.parse(xhr.response).filename;

var width = JSON.parse(xhr.response).width;

var height = JSON.parse(xhr.response).height;

console.log(xhr.response);

if(!JSON.parse(xhr.response).error){

document.getElementById("resp").innerHTML = "<strong>Ответ на запрос:</strong><br /><br />" +

"Название файла: <strong>" + filename + "</strong><br />" +

"Ширина изображения: <strong>" + width + "</strong><br />" +

"Высота изображения: <strong>" + height + "</strong><br />";

document.getElementById("myimage").innerHTML = '<img src="' + JSON.parse(xhr.response).imgURL + '" alt="" width="400">';

}

else{

document.getElementById("resp").innerHTML = "<strong>" + JSON.parse(xhr.response).error + "</strong>";

document.getElementById("myimage").innerHTML = "";

}

};

});

</script>

<body>

Лабораторная работа 6. Реализация REST API

<br /><br />

<strong>Поиск по содержимому изображения</strong><br /><br />

<form action="/byContent" method="post">

Название животного: <input type="text" id="animal" name="UI" type="text" placeholder="" size="40"></input><br /><br />

Цвет\*: <input type="text" id="colour" name="UI" type="text" placeholder="" size="40"></input><br /><br />

<input name="grouping" type="radio" value="single"> Одиночное животное\*

<input name="grouping" type="radio" value="multiple"> Группа животных\*<br /><br />

<input class = "sndCnt" type="submit" value="Submit" />

</form>

<strong>Поиск по содержимому изображения</strong><br /><br />

<form action="/byImage" method="post">

Имя изображения: <input type="text" id="filename" name="UI" type="text" placeholder="" size="40"></input><br /><br />

Ширина изображения: <input type="text" id="width" name="UI" type="text" placeholder="" size="10"></input><br /><br />

Высота изображения: <input type="text" id="height" name="UI" type="text" placeholder="" size="10"></input><br /><br />

<input class = "sndImg" type="submit" value="Submit" />

</form>

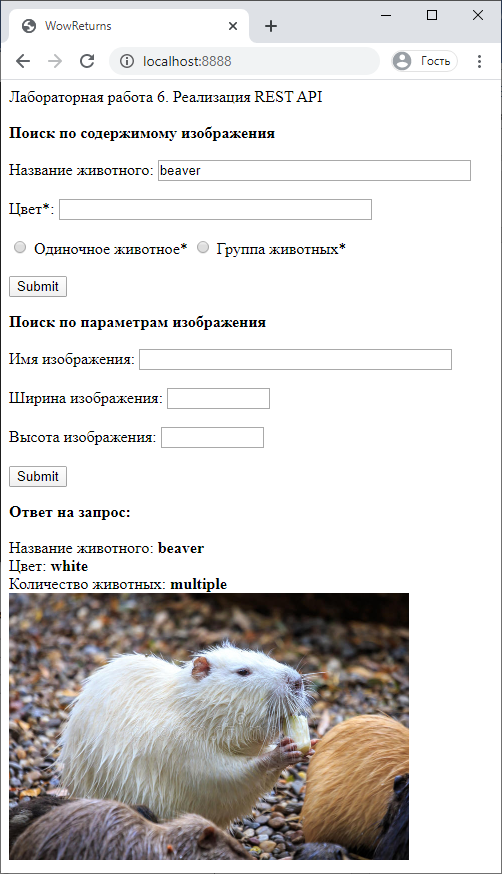
<a id = 'resp'></a>

<div id="myimage"></div>

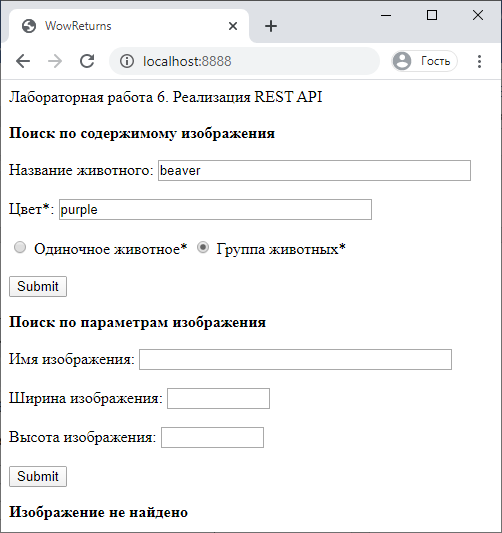
</body>

</html>

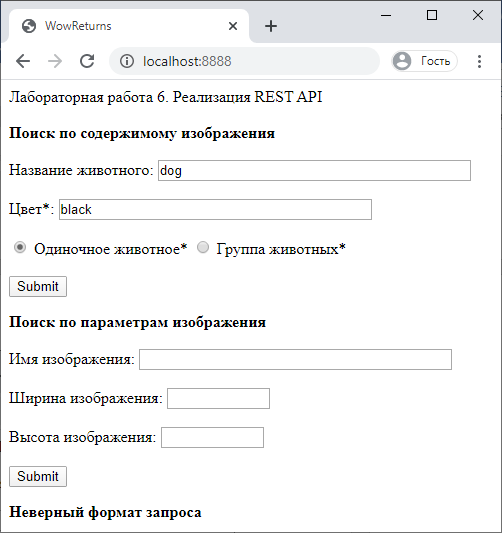
Результат работы программы показан на рисунках:



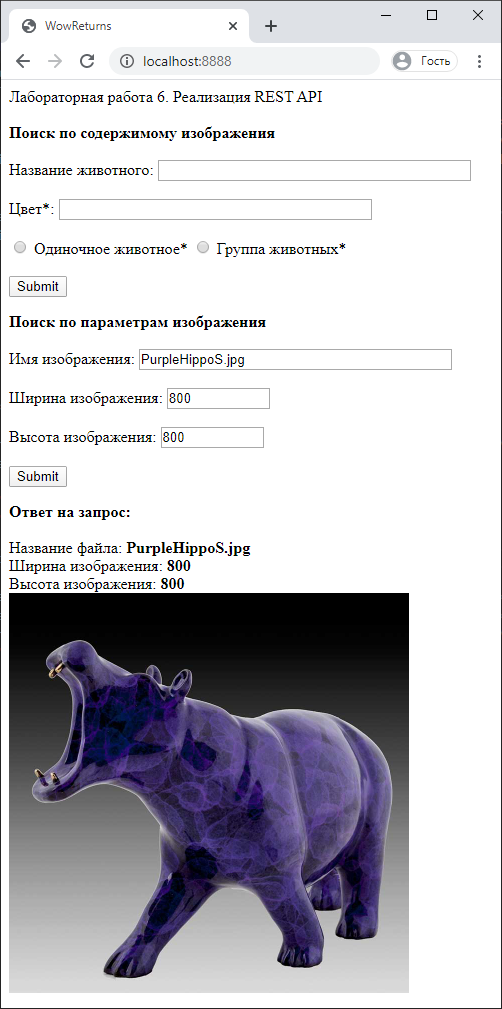
**Рисунок 1 – Запрос по содержимому изображения и ответ на запрос**



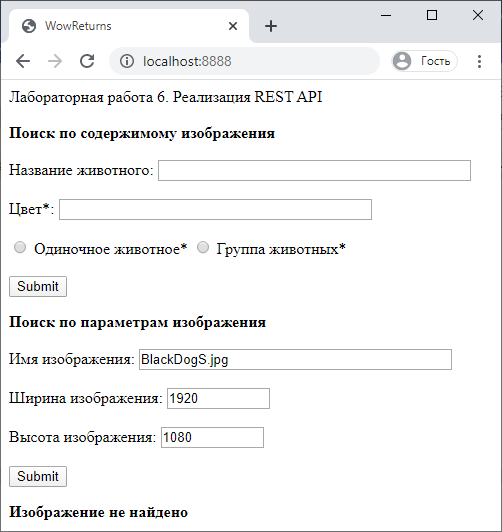
**Рисунок 2 – Запрос по содержимому изображения и ответ на запрос**



**Рисунок 3 – Запрос по содержимому изображения c неверно заполненным обязательным полем**



**Рисунок 4 – Запрос по параметрам изображения и ответ на запрос**



**Рисунок 5 – Запрос по параметрам изображения и ответ на запрос**

**Вывод**

В ходе выполнения лабораторной работы мы изучили основы обработки запросов клиента, реализовав в ней поддержку REST API.