Московский государственный университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Лабораторные работы по курсу:

«Разработка Интернет Приложений»

ЛР8. Javascript

Исполнитель:

Студент групп

РТ5-51

Кузьмин АС.

Преподаватель:

Гапанюк Ю. Е.

«___»_____

Москва 2017 г.

Задание и порядок выполнения

Разработать приложение для построения графиков тригонометрических функций на языке Javascript с HTML интерфейсом.

Ход работы:

1. Ознакомиться с теоретической частью
2. Создайте новый проект PyCharm
тип проекта: Pure Python
(мы не будем использовать Python в этой работе, просто это позволяет создать абсолютно пустой проект без зависимостей)
3. Добавьте в проект 2 файла:
a. index.html
b. index.js
4. Сверстайте страницу со следующими элементами:
a. два поля ввода для области определения аргумента (<input>)
b. поле для ввода функции (<input>)
c. кнопка “Построить график” (<button>)
d. поле вывода графика ( )
5. При помощи css укажите размеры блока графика, отличные от нуля
6. Присвойте каждому полю уникальный class (например, from, to, fun, output и т.д.)
7. Убедитесь, что ваша страница отображается в браузере нормально
8. Подключите jQuery, flot и ваш скрипт в index.html, используя теги <script>
<script src= "https://code.jquery.com/jquery-2.2.4.min.js" ></script>
<script src= "https://cdnjs.cloudflare.com/ajax/libs/flot/0.8.3/jquery.flot.js" ></script>
9. Переходим к разработке скрипта
10. Дождитесь загрузки страницы
```
$(function() {
// …
})
```
11. Найдите все элементы управления на вашей странице
```
var $from = $(‘.from);
```
12. Подпишитесь на событие нажатия кнопки
```
$button.click(onClick);
```
13. Отмените действие по-умолчанию (отправку формы)
```
e.preventDefault()
```
14. Получите значения из полей ввода
```
$from.val()
```
15. Не забудьте преобразовать числовые значения из строк в числа
parseFloat, parseInt
16. Создайте массив пар значений
```
const points = [[x1, y1], …, [xn, yn]];
```
17. Для того, чтобы получить значение функции, заданной в виде строки, используйте функцию eval()
```
const x = 0.1 ;
const fun = 'Math.sin(x)' ;
const y = eval (fun);
```
18. Постройте график по точкам
```
$. plot ( $ output, [ points ], {});
```
19. Проверьте правильность работы приложения, в случае проблем, воспользуйтесь отладчиком Chrome DevTools
20. Проверьте построение графиков функций:
a. Math.sin(x)
b. Math.random()
c. Math.exp(x)

21. Выведите название построенной функции в легенду:
http://www.flotcharts.org/flot/examples/basic-options/index.html
22. Дополнительное задание:
сделайте анимацию графика функции как на осциллографе
для этого по таймеру setInterval() / clearInterval() перестраивайте график
функции, прибавляя к ***x*** изменяющийся коэффициент ***dx***

Исходный код:

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <script src= "jquery-3.2.1.min.js"></script>
   <script src= "https://cdnjs.cloudflare.com/ajax/libs/flot/0.8.3/jquery.flot.js"></script>
   <script src="index.js"></script>
   <link rel="stylesheet" href="index.css">
   <meta charset="UTF-8">
   <title>Построение графиков</title>
</head>
<body>
   <br>
   <div class="body1">
      <p>From:</p>
      <p>To:</p>
      <p>Fun:</p>
   </div>
   <div class="body2">
      <p><input type="text" class="from" size="10" value="0"></p>
      <p><input type="text" class="to" size="10" value="35"></p>
      <p><input type="text" class="fun" size="10" value="Math.sin(x)*Math.exp(x/10)"></p>
      <button>Plot!</button>
   </div>
   <div class="center">
      <div class="coordinates_y">
         <p class="py1"></p><p class="py2"></p><p class="py3"></p><p class="py4"></p><p
class="py5"></p>
      </div>
      <canvas id="canvas" width="400" height="400"></canvas>
   </div>
   <div class="coordinates_x">
      <p class="px1"></p><p class="px2"></p><p class="px3"></p><p class="px4"></p><p
class="px5"></p>
   </div>
</body>
</html>
```

index.css

```
.body1 {
    vertical-align: top;
    margin: 0;
    width: 49%;
    display: inline-block;
    text-align: right;
}

.body2 {
    vertical-align: top;
    margin: 0;
    width: 10%;
    display: inline-block;
    text-align: left;
}

.body1 p {
    width: 100%;
    margin: 1px;
    height: 22px;
}

.body2 p {
    width: 100%;
    margin: 0px;
    height: 22px;
}

input {
    float: right;
    width: 95%;
    margin: 0;
    padding: 0;
}

button {
    float: right;
    border-color: grey;
    background-color: lightslategrey;
    border-radius: 2px;
    color: white;
}

#canvas {
    border: solid;
```

```
    border-width: 1px;
}

.center {
    display: flex;
    align-items: center;
    justify-content: center;
}

.center p{
    margin: 0px;
    padding-right: 3px;
}

.py1, .py2, .py3, .py4, .py5 {
    height: 100px;
    display: flex;
    align-items: center;
    justify-content: flex-end;
}

.coordinates_y {
    width: 6%;
}

.px1, .px2, .px3, .px4, .px5 {
    width: 100px;
    display: inline-block;
    text-align: center;
    margin: 0;
}

.coordinates_x {
    width: 94%;
    margin-left: 6%;
    display: flex;
    align-items: center;
    justify-content: center;
    position: relative;
    top: -47px;
}
```

index.js

```
$(function () {
    var $from = $('.from');
    var $to = $('.to');
    var $fun = $('.fun');
```

```
    var $button = $("button");
    clear_background();
    cell();
    $button.click(function(e) {
        e.preventDefault();
        var from = parseFloat($from.val());
        var to = parseFloat($to.val());
        var fun = $fun.val();
        const points = [];
        var miny = 0, maxy = 0;
        for (var x = from; x <= to; x = x+0.01) {
            var y = parseFloat(eval(fun));
            if (miny > y) miny = y;
            if (maxy < y) maxy = y;
            points.push([x.toFixed(2),y.toFixed(2)]);
        }
        var max = maxy;
        if (Math.abs(maxy) < Math.abs(miny)) max = Math.abs(miny);
        axes(max, from, to);
        plot(points, max);
        legend(fun);
    })
});

function plot(points, max) {
    clear_background();
    cell();
    var context = $("#canvas")[0].getContext("2d");
    var kx = context.canvas.width/(points.length-1);
    var ky = context.canvas.height/2/max;
    context.strokeStyle="#FF0000";
    context.lineWidth = 1;
    context.beginPath();
    context.moveTo(0, (context.canvas.width/2-ky*points[0][1]));

    // var i = 0, x = 0, y = 0;
    // var interval = setInterval(function() {
    //     x = i*kx;
    //     y = points[i][1];
    //     context.lineTo(x, (canvas.width/2-ky*y));
    //     context.stroke();
    //     if(i >= points.length-1) {
    //         clearInterval(interval);
    //     }
    //     i++;
    // }, 1);

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

```
        var x = i*kx;
        var y = points[i][1];
        context.lineTo(x, (context.canvas.width/2-ky*y));
    }
    context.stroke();
}

function cell()
{
    var context1 = $("#canvas")[0].getContext("2d");
    context1.strokeStyle="#000000";
    context1.lineWidth = 0.3;
    context1.beginPath();
    for (var i = 100; i <= 400; i = i + 100)
    {
        context1.moveTo(i,0);
        context1.lineTo(i,400);
    }
    for (var j = 100; j <=400; j = j + 100)
    {
        context1.moveTo(0,j);
        context1.lineTo(400,j);
    }
    context1.stroke();
}

function clear_background() {
    var context = $("#canvas")[0].getContext("2d");
    var my_gradient = context.createLinearGradient(0,0,0,400);
    my_gradient.addColorStop(0,"gainsboro");
    my_gradient.addColorStop(1,"white");
    context.fillStyle = my_gradient;
    context.fillRect(0, 0, context.canvas.width, context.canvas.height);
    //context.clearRect(0, 0, canvas.width, canvas.height);
}

function legend(fun_name)
{
    var context = $("#canvas")[0].getContext("2d");
    context.fillStyle = "black";
    context.font = "15pt Cambria Math";
    context.textAlign = "right";
    context.fillText(fun_name.replace(/Math./gi, ''), 400, 20);
}

function axes(max, from, to) {
    $('.py1').html(parseFloat(max).toFixed(1));
    $('.py2').html(parseFloat(max/2).toFixed(1));
```

```
    $('.py3').html(0);
    $('.py4').html(parseFloat(-max/2).toFixed(1));
    $('.py5').html(parseFloat(-max).toFixed(1));

    if (Math.abs(from) == Math.abs(to))
    {
        $('.px1').html(from.toFixed(1));
        $('.px2').html((from/2).toFixed(1));
        $('.px3').html(0);
        $('.px4').html((to/2).toFixed(1));
        $('.px5').html(to.toFixed(1));
    }
    else
    {
        $('.px1').html(from.toFixed(1));
        $('.px2').html((from+Math.abs(Math.abs(to)-Math.abs(from))/4).toFixed(1));
        $('.px3').html((from+Math.abs(Math.abs(to)-Math.abs(from))/2).toFixed(1));
        $('.px4').html((from+Math.abs(Math.abs(to)-Math.abs(from))/4*3).toFixed(1));
        $('.px5').html(to.toFixed(1));
    }
}
```

From: 0

To: 35

Fun: Math.sin(x)*Math.exp(x/10)

Plot!