Санкт-Петербургский государственный политехнический университет

Институт Прикладной математики и механики

Кафедра Прикладной математики

Т.А. Теницкая

МОДЕЛИРОВАНИЕ ТРЕБУШЕТА НА JAVASCRIPT

Курсовой проект

Направление подготовки бакалавров: 010800 Механика и математическое моделирование

Профиль ООП: 010800.62.01 Механика деформируемого твердого тела

Группа 23604/1

Руководитель проекта: Панченко А.Ю.

Допущен к защите:

«\_\_» 20\_\_ г.

Санкт-Петербург

2014

Оглавление

[**Введение** 3](#_Toc388981950)

[**Глава 1. Динамика требушета.** 4](#_Toc388981951)

[1.2. Динамика. 5](#_Toc388981952)

[**Глава 2. Написание кода** 6](#_Toc388981953)

[2.1. index.php 6](#_Toc388981954)

[2.2.Script.js 9](#_Toc388981955)

[2.3 Результат 21](#_Toc388981956)

[**Заключение** 23](#_Toc388981957)

[**Список литературы** 24](#_Toc388981958)

**Введение**

Человек всегда задумывался, как можно поразить добычу или врага на расстоянии. Для этого изобретались различные приспособления и орудия, однако, противник изобретал способы защиты и это противостояние продолжается по сегодняшний день. Данный курсовой проект посвящен моделированию одного из первых механизмов, предназначенных для поражения живой силы противника – требюше.

Требушет, также требюше (от фр. *trébuchet* — «вес ы с коромыслом») — метательная машина гравитационного действия для  осады городов. Первые сведения о машине, похожей по принципу действия на требушет, встречаются в Китае в V веке до н. э. в трудах древнекитайского философа  Мо Ди , однако нельзя сказать с уверенностью, что описанная им конструкция воспроизводит классический требушет. Остаётся неизвестным, когда требушеты появились в Европе. Первое известное упоминание о прототипе данной метательной машины содержит сочинение архиепископа фессалоникийского Иоанна «Чудеса святого Димитрия», составленное в 610—620 годах. Согласно историческим документам, начиная с 499 года, славяне вторгались и поселялись в Греции и в 580-х годах . Византия потеряла контроль почти над всем полуостровом. В течение VI —VII веков славяне несколько раз осаждали византийский город Фессалоники. Согласно свидетельству архиепископа Иоанна при осаде 597 года применялись передвижные осадные башни (гелеполы), «бараны» (тараны), камнемёты («огромнейшие», по словам Иоанна) и «черепахи». Нападавшие за сутки к утру 25 сентября изготовили на месте от 50 до 150 коромысловых камнемётов (по́роки ), которые, уходя, бросили, не считая особой ценностью. Вся эта военная техника, судя по всему, представляла собой заимствования из арсенала китайской полиоркетики, вероятно — через посредство тюрок или народов Средней Азии.

Среди всех средневековых метательных устройств требюше занимает особое, самое почетное место, благодаря своей простоте, надежности, грациозности и зрелищности. Поэтому неудивительно, что многие реконструкторы средневековья, да и просто любители военной техники в последнее время стали массово воспроизводить данное устройство, начиная от настольных моделей и заканчивая полноразмерными многотонными изделиями.

В курсовом проекте рассмотрен принцип работы требушета, найдены уравнения, описывающие его динамику и написан код на JavaScript , для моделирования данного устройства.

**Глава 1. Динамика требушета.**

**1.1 Принцип работы**

Требушет — гравитационная метательная машина (баробаллиста). Принцип его действия основан на использовании энергии падающего груза большой массы, закреплённого на коротком конце рычага. При этом длинный конец рычага разгоняет пращу со снарядом до большой скорости. Принцип действия простой, и для изготовления требушета необходима артель квалифицированных плотников, древесина подходящих размеров и знание относительно простой механики, что позволило требушетам вытеснить технически более сложные баллисты

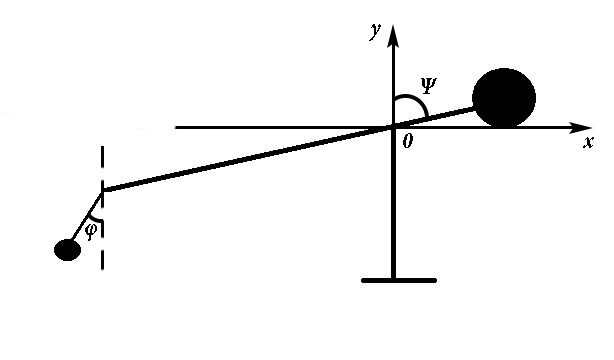
У простых требушетов тяжёлый противовес закреплён на рычаге и движется, соответственно, по дуге. У более технически совершенных машин груз подвешен на рычагах и падает вертикально вниз, более эффективно передавая снаряду гравитационную потенциальную энергию. Длина короткого и длинного метательного рычага по отношению к точке опоры должны соотноситься как 1:6, общая длина рычага определяется прочностью древесины и составляет около 10—13 м. Рычаг делался как из цельного дерева, так и из двух сложенных брусьев, скреплённых железными обручами.

Отличительной особенностью является механизм автоматического раскрытия пращи требушета, когда происходит выстрел и снаряд начинает самостоятельное перемещение. Длинный рычаг требушета соединен с двумя веревками пращи: один конец постоянно привязан к рычагу, второй в виде петли накидывается на конец рычага, на котором устроен крюк. Во время выстрела, когда праща набрала скорость, а перекладина с противовесом, проделав основной путь, начинает замедляться, концы пращи начинают скользить вокруг длинного конца с крюком, пока один из концов не соскакивает с крюка, вследствие чего праща раскрывается. Кривизна крюка и его длина определяют момент освобождения снаряда и, соответственно, траекторию его полета. Стреляющие могли корректировать дистанцию стрельбы и крутизну траектории (для стрельбы с горы, в гору, по прямой) с помощью специальных колец, укорачивающих крюк, либо заменяя сам крюк.

После выстрела взвод требушета осуществлялся обслугой, которая с помощью лебёдки притягивала верёвками длинный конец рычага вниз. Стрелять требушет мог всем, что помещалось в верёвочное «седло» («гнездо», «пяту») пращи: от каменных ядер и горшков с нефтью до отрубленных голов и живых людей.

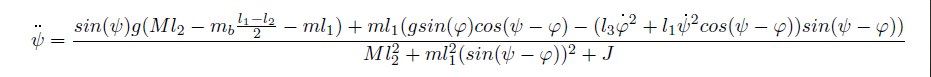
1.2. Динамика.

Будем рассматривать требушет, противовес которого закреплен на рычаге.



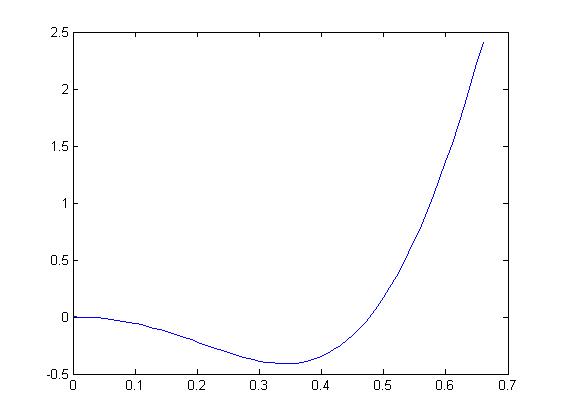
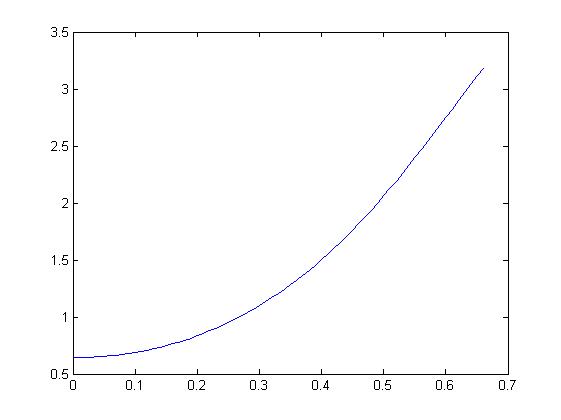
За обобщенные координаты выбраны угол отклонения от вертикали ядра с пращей и противовеса φ и ψ соответственно.

Записав уравнение Лагранжа 2-го рода, получена система:



http://cs614929.vk.me/v614929002/d2a7/-uvNY6rE8QM.jpg

Решили данную систему с помощью численного интегрирования. За условие отрыва возьмем ψ≈π. При  ψ = 3.18 φ = 2.41 скорость ядра Vm = 17.6  м/c. Ниже представлены графики ψ(t) и φ(t) соответственно:



**Глава 2. Написание кода**

2.1. index.php

Для начала создадим стартовую страницу index.php на которую и будем выводить наш фрейм для моделирования, график и слайдеры настройки.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Требушет</title>

<script src="ocanvas.min.js"></script>

<script src="script.js"></script>

<script src="TM.js"></script>

<script src="jquery-1.6.1.min.js"></script>

<script language="javascript" type="text/javascript " src="flot/jquery.flot.js"></script>

<script language="javascript" type="text/javascript " src="jquery.flot.axislabels.js"></script>

</div>

</head>

<body>

<div style="float:left; width:500px">  
<!--Слайдер для массы груза-->  
 <input type="range" id="slider\_M" style="width: 150px;"oninput="app.setSlider\_M(this.value ); document.getElementById('text\_M ').value = this.value;">

M =

<input id="text\_M" style="width: 5ex;" required pattern="[-+]?( [0-9]\*\.[0-9]+|[0-9]+)" oninput="

if (!this.checkValidity()) return;

app.setSlider\_M(this.value);

document.getElementById('slider\_M ').value = this.value ;"> кг  
<!--Кнопка «Старт»-->

<input type="button" style="width: 100px; margin-left:10px " name="" onclick="app.Start();" value="Start">

<br >   
<!--Слайдер для массы ядра-->

<input type="range" id="slider\_m" style="width: 150px;"oninput="app.setSlider\_m(this.value ); document.getElementById('text\_m ').value = this.value;">

m =

<input id="text\_m" style="width: 5ex;" required pattern="[-+]?( [0-9]\*\.[0-9]+|[0-9]+)" oninput="

checkValidity( ) вернет false

if (!this.checkValidity()) return;

app.setSlider\_m(this.value);

document.getElementById('slider\_m ').value = this.value ;"> кг  
<!--Кнопка «Рестарт»-->

<input type="button" style="width: 100px; margin-left:10px " name="" onclick="app.ReStart();" value="Restart">

</div>

<!--График скорости-->

<div id="vGraph" style="width:500px; height:150px; margin-left: 450px">

</div>

<div style="float:left; width: 1200px">  
<!--Канвас-->

<canvas id="OCanvas" width="1200" height="450" </canvas>

</div>

<!—Передача в скрипт ссылок на канвас, слайдеры, textarea-->

<script type="text/javascript">var app = new Func(

document.getElementById('OCanvas'),

document.getElementById('slider\_M'),

document.getElementById('text\_M'),

document.getElementById('slider\_m'),

document.getElementById('text\_m'),

document.getElementById('photo')

);</script>

</body>

</html>

Для создания меню используем библиотеку jQuery . oCanvas нам нужно для вывода анимации требушета.

Далее создадим скрипт, в котором и будет описано движение ядра.

2.2.Script.js

function Func (canvas, slider\_M , text\_M , slider\_m , text\_m, photo) {

var canvas = oCanvas.create({

canvas: "#OCanvas",

fps:48

});

var kal = 0, x\_fly , y\_fly;

var x \_2, y\_2;

var M = 1000; //Задаем массу противовеса

var m = 100; //Задаем массу ядра

var vx = 0;

var vy = 0;

var go = 0;

var L\_1 = 1; //Длина большего плеча рычага

var L\_2 = 0.5; //Длина меньшего плеча рычага

var L\_3 = 0.6; //Длина пращи

var L = L \_1 + L\_2;

var g = 9.8;

var L\_12 = (L\_1-L\_2)/2;

var v\_m = 0;

var m\_beam = 100; //Длина рычага

var psi = []; // Задаем массив для ψ

var phi = []; //Задаем массив для φ

var psi\_diff = []; //Задаем массив для первой производной ψ

var phi\_diff = []; // Задаем массив для первой производной φ

var psi\_ddiff = []; // Задаем массив для второй производной ψ

var phi\_ddiff = []; // Задаем массив для второй производной φ

var max = 0;

var max\_n = 0;

var x = 0;

var y = 0;

var dt = 0;

var g = 9.81;

var what = false;

var J = (m\_beam \* L \* L)/12 + m\_beam \* L\_12 \* L\_12; //Момент инерции рычага

var t = 0;

var tmax = 5; //Время

var dt = tmax/1000;

var n = (tmax/dt+1);

var t = [];

t[0] = 0;

this.setSlider\_M = function(new\_M ){M = new\_M;}; //Задание M и m при перемещении слайдера

this.setSlider\_m = function(new\_m ){m = new\_m;};

this.Start = function(){calculations(); what = true; max \*= 5, t = 0;}; //Функция, по нажатию кнопки «Старт»

this.ReStart = function(){canvas.display.rectangle({ //Функция по нажатию кнопки «Рестарт»

x: 400,

y: 170,

width: 300,

height: 200,

fill: "#fff"

}).add();what = false; trajectory = []; ReDraw(); kal = 0; dt = tmax/1000;};

slider\_M.min = 100; slider\_M.max = 10000;

slider\_M.step = 10;

slider\_M.value = M ; // начальное значение ползунка должно задаваться после min и max

text\_M.value = M;

slider\_m.min = 10; slider\_m.max = 500;

slider\_m.step = 1;

slider\_m.value = m ; // начальное значение ползунка должно задаваться после min и max

text\_m.value = m;

function calculations() { //Расчет ψ, φ и максимальной скорости ядра

max = 0;

go = 0;

go = 0;

for(var k = 1; k < n; k ++)

{

t[k ] = t[k -1] + dt;

}

psi[ 0] = Math.acos((0.4)/(0.5)); //Начальные условия для ψ и φ

phi[0] = 0;

psi\_diff[0] = 0;

phi\_diff[0] = 0;

//Уравнение для второй производной ψ

psi\_ddiff[0] = (Math.sin(psi[0])\*g\*(M\*L\_2-m\_beam\*L\_12-m\*L\_1)+

m\*L\_1\*( g\*Math.sin(phi[0])\*Math.cos(psi[0]-phi[0]) -

L\_3\*Math.pow(phi\_diff[0],2)\*Math.sin(psi[0]-phi[0]) -

L\_1\*Math.pow(psi\_diff[0],2)\*Math.cos(psi[0]-phi[0])\*Math.sin(psi[0]-phi[0]) ) ) / (

M\*Math.pow(L\_2,2) + m\*Math.pow(L\_1,2)\*Math.pow(Math.sin(psi[0]-phi[0]),2) + J );

//Уравнение для второй производной φ

phi\_ddiff[ 0] = ( L\_1\*( Math.pow(psi\_diff[0],2)\*Math.sin(psi[0]-phi[0]) -

psi\_ddiff[0]\*Math.cos(psi[0]-phi[0]) ) - g\*Math.sin(phi[0]) ) / L\_3;

//Численное интегрирование методом конечных разностей

for(var k = 1; k < n ; k ++)

{

psi[ k] = psi[k-1] + psi\_diff[k-1]\*dt;

phi[ k] = phi[k-1] + phi\_diff[k-1]\*dt;

psi\_diff [k] = psi\_diff[ k-1] + psi\_ddiff[k-1]\*dt;

phi\_diff [k] = phi\_diff[ k-1] + phi\_ddiff[k-1]\*dt;

psi\_ddiff[k] = (Math.sin(psi[k])\*g\*(M\*L\_2-m\_beam\*L\_12-m\*L\_1)+

m\*L\_1\*( g\*Math.sin(phi[k])\*Math.cos(psi[k]-phi[k]) -

L\_3\*Math.pow(phi\_diff[k],2)\*Math.sin(psi[k]-phi[k]) -

L\_1\*Math.pow(psi\_diff[k],2)\*Math.cos(psi[k]-phi[k])\*Math.sin(psi[k]-phi[k]) ) ) / (

M\*Math.pow(L\_2,2) + m\*Math.pow(L\_1,2)\*Math.pow(Math.sin(psi[k]-phi[k]),2) + J );

phi\_ddiff[k] = (L\_1\*psi\_diff[k]\*psi\_diff[k]\*Math.sin(psi[k]-phi[k]) - g\*Math.sin(phi[k]) - L\_1\*psi\_ddiff[k]\*Math.cos(psi[k]-phi[k]))/L\_3;

}

while (psi[go] <= 4)

{

// Расчет скорости ядра

v\_m = Math.sqrt ( Math.pow(L\_1\*psi\_diff [go],2) + Math.pow(L\_3\*phi\_diff[go],2) + 2\*L\_1\*L\_3\*psi\_diff[go]\*phi\_diff[go]\*Math.cos(psi[go]-phi[go]));

if (v\_m > max)

{

max = v\_m;

max\_n = go;

}

go++;

}

v = max;

};

//Высота отрыва ядра

var H = -L\_1 \* Math.cos(psi[max\_n ]) - L\_3 \* Math.cos(phi[max\_n]);

var beta = -90;

var V\_m;

var ffd = Math.sqrt((M \* 9.81 \* 2 - m \* 9.81 \* 8) / ((M \* 8 + m / 2 )));

function ReDraw(){ //Функция для перерисовки

line.start.x = 250; / / Координаты , задающие положение рычага, пращи, груза и ядра

line.end.x = 210;

line.start.y = 300;

line.end.y = 300;

line\_1.start.x = 50;

line\_1.end.x = 210;

line\_1.start.y = 300;

line\_1.end.y = 300;

cargo.x = line.start.x;

cargo.y = line.start.y;

sling.start.x = 50;

sling.end.x = 50;

sling.start.y = 300;

sling.end.y = 330;

ball.x = sling.end.x;

ball.y = sling.end.y

};

var line\_d = canvas.display.line({

start: { x: 0, y: 450 },

end: { x: 2000, y: 450 },

stroke: "2px #000",

cap: "round"

}).add();

//Задание рычага

var line = canvas.display.line({

start: { x: 250, y: 300 },

end: { x: 210, y: 300 },

stroke: "2px #000",

cap: "round"

}).add();

var line\_1 = canvas.display.line({

start: { x: 50, y: 300 },

end: { x: 210, y: 300 },

stroke: "2px #000",

cap: "round"

}).add();

var line\_2 = canvas.display.line({

start: { x: 210, y: 300 },

end: { x: 210, y: 450 },

stroke: "2px #000",

cap: "round"

}).add();

//Задание картинки замка

var image = canvas.display.image({

x: 1100,

y: 310,

origin: { x: "center", y: "center" },

image: "castle.png"

}).add();

//Рисование груза

var cargo = canvas.display.ellipse({

x:line.start.x,

y:line.start.y,

radius:15,

fill: "rgba(255, 0, 0, 1)"

}).add();

//Рисование пращи

var sling = canvas.display.line({

start: { x: 50, y: 300 },

end: { x: 50, y: 330 },

stroke: "2px #000",

cap: "round"

}).add();

//Рисование ядра

var ball = canvas.display.ellipse({

x:sling.end.x,

y:sling.end.y,

radius:5,

fill: "rgba(0, 0, 255, 1)"

}).add();

var trajectory = [];

//Функция для движения рычага с грузами

function moveR()

{

if (what && ball.y <=450)

{

x1 = line.end.x;

y1 = line.end.y;

x2 = line\_1.end.x;

y2 = line\_1.end.y;

x3 = sling.end.x;

y3 = sling.end.y;

Radius = 40;

Radius\_1 = 160;

Radius\_2 = 30;

x = parseInt( Radius \* Math.cos(psi[kal]-Math.PI /2) + x1);

y = parseInt( Radius \* Math.sin(psi[kal]-Math.PI/2) + y1);

x\_1 = parseInt( Radius\_1 \* Math.cos(psi[kal ] + Math.PI /2) + x2);

y\_1 = parseInt( Radius\_1 \* Math.sin(psi[kal ] + Math.PI/2) + y2);

x\_2 = parseInt( Radius\_2 \* Math.cos(phi[kal ] + Math.PI/2) + x3);

y\_2 = parseInt( Radius\_2 \* Math.cos(phi[kal ] + Math.PI/2) + y3);

x\_y(x, y, x\_1, y\_1, x\_2, y\_2)

kal++;

if (kal >= max\_n)

{

x0 = x\_2;

y0 = y\_2;

fly();

x\_y\_after(x, y, x\_1, y\_1, x\_2, y\_2);

}

};

}

//Функция, отвечающая за полет ядра

function fly()

{

t += tmax/1000;

x\_fly = x0 + max \* Math.cos(-Math.PI /4) \* dt;

y\_fly = y0 + max \* Math.sin(-Math.PI /4) \* dt + g \* dt \* dt / 2;

vx = v \* Math.cos(Math.PI/4);

vy = v \* Math.sin(Math.PI/4) - g \* t;

//График скорости

trajectory.push( [t, Math.sqrt(Math.pow(vy,2)+Math.pow(vx,2))]);

$.plot($('#vGraph'), [trajectory], {});

dt += 0.1

ball.x = x\_fly;

ball.y = y\_fly;

if ((x\_fly > 1000 && y\_fly > 159) && (x\_fly < 1200 && y\_fly < 450))

{

var text = canvas.display.text({

x: 600,

y: 225,

origin: { x: "center", y: "top" },

font: "bold 30px sans-serif",

text: "Вы попали!",

fill: "#a00"

}).add();

};

}

//Функция перерисовки требушета с грузом и ядром

function x\_y (x,y, x\_1, y\_1, x\_2, y\_2){

line.start.x = x;

line.start.y = y;

line\_1.start.x = x\_1;

line\_1.start.y = y\_1;

cargo.x = x;

cargo.y = y;

ball.x = x\_2;

ball.y = y\_2;

sling.end.x = x\_1;

sling.end.y = y\_1;

sling.start.x = x\_2;

sling.start.y = y\_2;

}

//Функция перерисовки требушета после вылета ядра

function x\_y\_after (x,y, x\_1, y\_1, x\_2, y\_2){

line.start.x = x;

line.start.y = y;

line\_1.start.x = x\_1;

line\_1.start.y = y\_1;

cargo.x = x;

cargo.y = y;

sling.end.x = x\_1;

sling.end.y = y\_1;

sling.start.x = x\_2;

sling.start.y = y\_2;

}

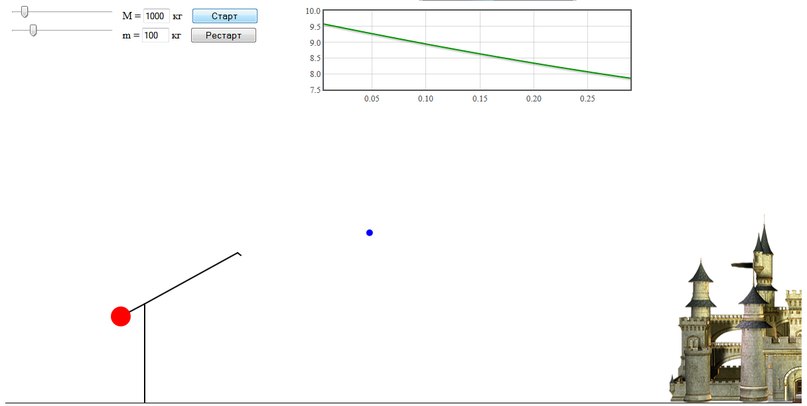
canvas.setLoop(moveR).start();

}

2.3 Результат

Итогом программы стала игра «Требушет», в которой с помощью меню можно задавать массы ядра и противовеса. Если ядро долетает до замка - игра пройдена, если нет – провалена.

Скриншоты из игры: 

Запускаем игру:

Попадение в замок:



# Заключение

В ходе написания курсовой работы был прочитан и систематизирован теоретический материал по данной теме, а так же выполнены следующие поставленные во введении задачи:

* Изучено устройство и принцип работы требушета
* Выведены уравнения для решения задачи динамики требушета
* Найдена скорость, с которой ядро вылетает из пращи
* Написана простая игра на JavaScript
* Каждый раз при запуске игры строится график зависимости скорости ядра от времени

Изучив материал по использованию JavaScript, я пришла к выводу, что для наглядного мо делирования JavaScript достаточно удобен, так как он обладает широкими возможностями и простым синтаксисом .

**Список литературы**

1. http://ru.wikipedia.org/wiki/Библиотека\_JavaScript
2. http://jqbook.net.ru/
3. http://ru.html.net/
4. Антон Шевчук. Jquery. Учебник для начинающих.: 2013. – 123с.
5. Бенедетти Р., Крэнли Р. Изучаем работу с jQuery. – СПб.: Питер, 2012. – 528 с.
6. БерБибо, Иегуда Кац. jQuery. Подробное руководство по продвинутомуJavaScript, 2-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 624 с.
7. Каслдайн Э., Шарки К. Изучаем jQuery. 2-е изд. – СПб.: Питер, 2012. – 400 с.