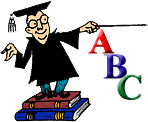
**Применение JavaScript в Web-дизайне**



**Для того чтобы сайт пользовался популярностью, он должен иметь оригинальный дизайн, какие-либо визуальные эффекты и при этом быстро загружаться. Это можно осуществить, используя язык JavaScript. С помощью этого языка программирования можно создавать визуальные эффекты на Web-странице, например, меняющийся цвет фона, движение, а также использовать сценарии для размещения элементов страницы Рассмотрим эти примеры подробнее.  
        Пример1. Эффект печатной машинки. Принцип работы сценария – последовательное извлечение из заданной строки определенного символа. Если количество извлекаемых символов совпадает с длинной строки, то поле очищается, и процесс извлечения символов начинается сначала. Рассмотрим код этой Web-странички:  
<HTML>  
<HEAD>  
<TITLE> Эффект печатной машинки </TITLE>  
<script language="JavaScript">  
var line="Java - клёвая штука !"; // Задаем строковую переменную  
var speed=150; // Устанавливаем скорость появления символов  
var i=0; // начальное количество символов, извлекаемых из строки  
function m\_line() // Создаем функцию вывода символов  
{   
if(i++<line.length)   
{  
document.cit.forum.value=line.substring(0,i);   
}else  
{  
document.cit.forum.value=" ";   
i=0;   
}setTimeout('m\_line()',speed);}  
</script>  
<center>  
<form name=cit>  
<input type=text size=32 name=forum>  
</form>  
</center>  
<script language="JavaScript">m\_line();</script>  
</HEAD>  
<BODY TEXT="#000000" BGCOLOR="#FFFFFF" BACKGROUND=""   
</BODY>  
</HTML>  
Задание.Наберите этот пример в текстовом редакторе и сохранить в файле с именем des1.html.**[**Откройте**](http://jscriptforschool.narod.ru/frame_javas/des1.html)**этот файл в браузере. Попробуйте создать свою бегущую строку, указав в тэге<marquee> свои параметры.**[**Просмотрите**](http://jscriptforschool.narod.ru/frame_javas/des1.html)**созданный файл в Internet-браузере. На страничке должно появиться следующее:**

Начало формы



Конец формы

**Пример 2. Мигание экрана. Принцип действия сценария – чередование цвета фона через определенный промежуток времени. Рассмотрим код этогой странички:  
<html>  
<head>  
<title>Мигание экрана</title>  
<script language="JavaScript">  
<!--   
function blinkit() //создаем функцию, позволяющую экрану мигать  
{   
intrvl=0; // начальный интервал перед изменением цвета фона  
for(nTimes=0;nTimes<3;nTimes++)   
{  
intrvl += 200; // интервал перед изменением цвета фона  
setTimeout("document.bgColor='#0000FF';",intrvl);  
intrvl += 200; // интервал перед изменением цвета фона  
setTimeout("document.bgColor='#FFFFFF';",intrvl);  
}  
}  
// -->  
</script>  
</head>  
<body>  
<div align="center">  
<h3> При нажатии на кнопку экран будет мигать. </h3>  
<form>  
<input type="button" value="Мигание экрана" onClick="blinkit()"> // создание кнопки  
</form>  
</div>  
</body>  
</html>  
Просмотрите созданный файл в Internet-браузере. В результате выполнения кода получится следующая страница:**

**Задание.Наберите этот пример в текстовом редакторе и сохранить в файле с именем des2.html.**[**Откройте**](http://jscriptforschool.narod.ru/frame_javas/des2.html)**этот файл в браузере. Измените сценарий так, чтобы фон странички мигал тремя цветами.**

**Пример 3. Приближение и удаление текста. В основе сценария – изменение размера шрифта строки (чередующееся увеличение и уменьшение). Создайте HTML – документ des3.html, содержащий следующий код:  
  
<html>  
<script language="JavaScript">  
<!--   
  
var speed = 100; // скорость приближения  
var cycledelay = 2000; //значение задержки между окончанием приближения и началом удаления   
  
текста  
var maxsize = 48; // максимальный размер шрифта  
// описание переменных  
var x = 0;   
var y = 0;  
var themessage, size;  
var esize = "</font>";  
  
function initArray() // функция определяющая параметры строки  
{  
this.length = initArray.arguments.length;  
for (var i = 0; i < this.length; i++) {  
this[i] = initArray.arguments[i];  
}  
}  
var themessage2 = new initArray("Приближение и удаление текста");  
if(navigator.appName == "Netscape")  
document.write('<layer id="wds"></layer><br>');  
if (navigator.appVersion.indexOf("MSIE") != -1)  
document.write('<span id="wds"></span><br>');  
  
function upwords() // функция приближения строки  
{   
themessage = themessage2[y];  
if (x < maxsize) {  
x++;  
setTimeout("upwords()",speed);  
}  
else setTimeout("downwords()",cycledelay);  
  
if(navigator.appName == "Netscape") {  
size = "<font point-size='"+x+"pt'>";   
document.wds.document.write(size+"<center>"+themessage+"</center>"+esize);  
document.wds.document.close();  
}  
if (navigator.appVersion.indexOf("MSIE") != -1){  
wds.innerHTML = "<center>"+themessage+"</center>";  
wds.style.fontSize=x+'px';  
}  
}   
function downwords() //функция удаления строки  
{  
if (x > 1) {  
x--;  
setTimeout("downwords()",speed);  
}  
else {  
setTimeout("upwords()",cycledelay);  
y++;  
if (y > themessage2.length - 1) y = 0;  
}  
if(navigator.appName == "Netscape") {  
size = "<font point-size='"+x+"pt'>";   
document.wds.document.write(size+"<center>"+themessage+"</center>"+esize);  
document.wds.document.close();  
}  
if (navigator.appVersion.indexOf("MSIE") != -1){  
wds.innerHTML = "<center>"+themessage+"</center>";  
wds.style.fontSize=x+'px';  
}  
}  
setTimeout("upwords()",speed);// задержка между окончанием приближения и началом удаления   
  
текста  
// End -->  
</script>  
<head>  
<title> Приближение и удаление текста </title>  
</head>  
<body>  
</body>  
</html>  
На получившейся Web-страничке мы можем увидеть следующее содержимое:**

**А  следующий пример покажет как можно использовать сценарии для работы с изображениями. В этом примере сценарий позволяет задавать положение изображений относительно текста. рассмотрим код этого документа:**

**<html>   
<head>  
<title>Расположение изображения относительно строки</title>   
<script language="JavaScript">  
<!--  
var k1="TOP"   
var k2="TOP"   
var k3="TOP"   
function setk(i)   
{var k  
var obj=eval("form"+i)   
if(((obj.elements[0])[0]).selected)  
k=((obj.elements[0])[0]).value   
if (((obj.elements[0])[1]).selected)  
k=((obj.elements[0])[1]).value   
if(((obj.elements[0])[2]).selected)  
k=((obj.elements[0])[2]).value   
if(((obj.elements[0])[3]).selected)  
k=((obj.elements[0])[3]).value   
if(((obj.elements[0])[4]).selected)  
k=((obj.elements[0])[4]).value   
if(((obj.elements[0])[5]).selected)  
k=((obj.elements[0])[5]).value   
if(i==1)  
k1=k   
else  
if(i==2)   
k2=k   
else   
k3=k  
}  
function set()  
{document.pict1.align=k1   
{document.pict2.align=k2   
{document.pict3.align=k3  
}  
}  
}  
//-->  
</script>   
</head>  
<body bgcolor="f8f8ff">   
<center>  
<h4 align="center"> Pacпoлoжeниe изображения относительно строки </h4>  
Для трех используемых изображений выберите значение параметра выравнивания align и нажмите кнопку <b> Просмотр</b><br>  
Вы увидите как располагаются изображения в тексте.   
<table border=2 bgcolor=silver> <tr>  
<td align=center><img src="fon1.gif" width=75></td>   
<td align=center><img src="fon5.gif" width="69" height="71">   
<td align=center><img src="fon6.gif" width="72" height="73"> </td></tr>   
<tr>  
<td><form name="form1">   
<select name=sel1 size=1 onchange="setk(1)">   
<option value="top" checked>TOP   
<option value="texttop">TEXTTOP   
<option value="middle">MIDDLE   
<option value="left">LEFT   
<option value="bottom">BOTTOM   
<option value="right">RIGHT   
</select>   
</form></td>  
<td><form name="form2">  
<select name=sel2 size=l onchange="setk(2)">   
<option value="top" checked>TOP   
<option value="texttop">TEXTTOP   
<option value="middle">MIDDLE   
<option value="left">LEFT   
<option value="bottom">BOTTOM   
<option value="right">RIGHT   
</select> </form></td>   
<td><form name="form3">  
<select name=sel3 size=l onchange="setk(3)">   
<option value="top" checked>TOP   
<option value="texttop">TEXTTOP   
<option value="middle">MIDDLE   
<option value="left">LEFT   
<option value="bottom">BOTTOM   
<option value="right">RIGHT   
</select> </form></td> </tr></table>   
<form>  
<input type="button" value="Просмотр" onclick="set()">   
<input type="reset" value="Обновить"> </form> </center>  
<table border=2 bgcolor="#ffdcdc"> <tr><td>  
Параметры выравнивания   
<img src="fon1.gif" align=TOP name=pict1 width="67" height="58">задают   
<img src="fon5.gif" align=TOP name=pict2 width="63" height="58">   
расположение изображения относительно строки текста.   
<img src="fon6.gif" align=TOP name=pict3 width="64" height="62">  
В этом случае изображение является обычным элементом строки.  
</td></tr>  
</table>   
</body>   
</html>**

**В результате получиться следущее:**

**Pacпoлoжeниe изображения относительно строки**

**Для трех используемых изображений выберите значение параметра выравнивания align и нажмите кнопку Просмотр  
Вы увидите как располагаются изображения в тексте.**

|  |  |  |
| --- | --- | --- |
| **http://jscriptforschool.narod.ru/frame_javas/fon1.gif** | **http://jscriptforschool.narod.ru/frame_javas/fon5.gif** | **<http://jscriptforschool.narod.ru/frame_javas/fon6.gif>** |
| Начало формы    Конец формы | Начало формы    Конец формы | Начало формы    Конец формы |

Начало формы



Конец формы

* [Главная](http://ruseller.com/)»

* [Уроки](http://ruseller.com/lessons.php)»

* [jQuery](http://ruseller.com/lessons.php?rub=32)»

* Эффект приближения для миниатюр
* [Метки](http://ruseller.com/lessons.php?tag) урока:

* [jquery](http://ruseller.com/lessons.php?tag=8)

* [галерея](http://ruseller.com/lessons.php?tag=9)

* [кодинг](http://ruseller.com/lessons.php?tag=22)

* [javascript](http://ruseller.com/lessons.php?tag=24)

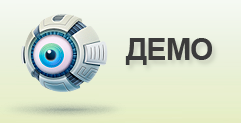
* [web дизайн](http://ruseller.com/lessons.php?tag=33)

[](http://1popov.ru/c/87585)

# Эффект приближения для миниатюр

Создадим эффект приближения для миниатюр с помощью jQuery. Масштаб миниатюр будет изменяться в зависимости от расстояния между ними и курсором мыши.

В уроке используется [плагин jQuery Proximity](https://github.com/padolsey/jQuery-Plugins/tree/master/proximity-event" \t "_blank), который написал [James Padolsey](http://james.padolsey.com/" \t "_blank).

[](http://ruseller.com/lessons/les1359/demo/index.html)[](http://ruseller.com/lessons/les1359/demo_1359.zip)

## Разметка HTML

Мы будем использовать неупорядоченный список миниатюр с секцией описания (в первой демонстрации не используется):

|  |  |  |
| --- | --- | --- |
| 01 | <ul id="pe-thumbs" class="pe-thumbs"> | |
| 02 | <li> |

|  |  |
| --- | --- |
| 03 | <a href="#"> |
| 04 | <img src="images/thumbs/1.jpg" /> | |

|  |  |
| --- | --- |
| 05 | <div class="pe-description"> |
| 06 | <h3>Заголовок описания</h3> | |

|  |  |  |
| --- | --- | --- |
| 07 | <p>Текст описания</p> | |
| 08 | </div></a> |

|  |  |
| --- | --- |
| 09 | </li> |
| 10 | <li><!-- ... --></li> | |

|  |  |
| --- | --- |
| 11 | </ul> |

## CSS

Неупорядоченный список будет центрироваться на странице и иметь тематическое фоновое изображение, просвечивающее через миниатюры с низким значение непрозрачности:

|  |  |
| --- | --- |
| 1 | .pe-thumbs{ |
| 2 | width: 900px; | |

|  |  |
| --- | --- |
| 3 | height: 400px; |
| 4 | margin: 20px auto; | |

|  |  |
| --- | --- |
| 5 | position: relative; |
| 6 | background: transparent url(../images/3.jpg) top center; | |

|  |  |
| --- | --- |
| 7 | border: 5px solid #fff; |
| 8 | box-shadow: 0 1px 2px rgba(0,0,0,0.2); | |

|  |  |
| --- | --- |
| 9 | } |

Мы также слегка затонируем фоновое изображение с помощью псевдо-элемента с фоновым цветом в формате *rgba*:

|  |  |  |
| --- | --- | --- |
| 01 | .pe-thumbs:before { | |
| 02 | content: ""; |

|  |  |
| --- | --- |
| 03 | display: block; |
| 04 | position: absolute; | |

|  |  |
| --- | --- |
| 05 | top: 0px; |
| 06 | left: 0px; | |

|  |  |
| --- | --- |
| 07 | width: 100%; |
| 08 | height: 100%; | |

|  |  |  |
| --- | --- | --- |
| 09 | background: rgba(255,102,0,0.2); | |
| 10 | } |

Список пунктов будет смещаться влево и мы установим относительное положение ссылок и изображений:

|  |  |
| --- | --- |
| 1 | .pe-thumbs li{ |
| 2 | float: left; | |

|  |  |  |
| --- | --- | --- |
| 3 | position: relative; | |
| 4 | } |

|  |  |
| --- | --- |
| 5 | .pe-thumbs li a, |
| 6 | .pe-thumbs li a img{ | |

|  |  |
| --- | --- |
| 7 | display: block; |
| 8 | position: relative; | |

|  |  |
| --- | --- |
| 9 | } |

Каждая миниатюра будет иметь начальную ширину 100 px и непрозрачность 0.2:

.pe-thumbs li a img{

    width: 100px;

    opacity: 0.2;

}

Описание будет скрыто по умолчанию и затем выводиться с помощью JavaScript:

|  |  |  |
| --- | --- | --- |
| 01 | .pe-thumbs li a div.pe-description{ | |
| 02 | width: 200px; |

|  |  |
| --- | --- |
| 03 | height: 100px; |
| 04 | background: rgba(0,0,0,0.8); | |

|  |  |  |
| --- | --- | --- |
| 05 | position: absolute; | |
| 06 | top: 0px; |

|  |  |
| --- | --- |
| 07 | left: -200px; |
| 08 | color: white; |

|  |  |
| --- | --- |
| 09 | display: none; |
| 10 | z-index: 1001; |

|  |  |  |
| --- | --- | --- |
| 11 | text-align: left; | |
| 12 | } |

В завершение определим стили для описания:

|  |  |
| --- | --- |
| 01 | .pe-description h3{ |
| 02 | padding: 10px 10px 0px 10px; | |

|  |  |
| --- | --- |
| 03 | line-height: 20px; |
| 04 | font-family: 'BebasNeueRegular', 'Arial Narrow', Arial, sans-serif; | |

|  |  |  |
| --- | --- | --- |
| 05 | font-size: 22px; | |
| 06 | margin: 0px; |

|  |  |
| --- | --- |
| 07 | } |
| 08 | .pe-description p{ | |

|  |  |  |
| --- | --- | --- |
| 09 | padding: 10px 0px; | |
| 10 | margin: 10px; |

|  |  |
| --- | --- |
| 11 | font-size: 11px; |
| 12 | font-style: italic; | |

|  |  |  |
| --- | --- | --- |
| 13 | border-top: 1px solid rgba(255,255,255,0.3); | |
| 14 | } |

## JavaScript

Сначала рассчитаем размер контейнер с описанием и его положение при наведении курсора мыши на миниатюру. Данные значения зависят от максимального масштаба миниатюры и ее расположения в основном контейнере. Например, когда миниатюра находится рядом с правым краем, описание должно появляться слева от миниатюры.

Затем надо привязать событие приближения к изображению. Масштаб изображения будет изменяться в соответствии с положением курсора мыши. Как только масштаб получит максимальное значение, установим большой *z-index*, чтобы миниатюра выводилась поверх всего, и выведем соответствующее описание:

|  |  |
| --- | --- |
| 001 | // Список миниатюр |
| 002 | var $list       = $('#pe-thumbs'), | |

|  |  |
| --- | --- |
| 003 | // Ширина и смещение влево списка. |
| 004 | // Данные значения будут использоваться для получения положения контейнера описания. | |

|  |  |
| --- | --- |
| 005 | listW       = $list.width(), |
| 006 | listL       = $list.offset().left, | |

|  |  |
| --- | --- |
| 007 | // Изображения |
| 008 | $elems      = $list.find('img'), | |

|  |  |
| --- | --- |
| 009 | // Контейнер описания |
| 010 | $descrp     = $list.find('div.pe-description'), | |

|  |  |
| --- | --- |
| 011 | // maxScale : максимально значение масштаба, которое может иметь изображение |
| 012 | // minOpacity / maxOpacity : минимальное (устанавливается в CSS) и максимальное значение непрозрачности изображения |

|  |  |
| --- | --- |
| 013 | settings    = { |
| 014 | maxScale    : 1.3, | |

|  |  |
| --- | --- |
| 015 | maxOpacity  : 0.9, |
| 016 | minOpacity  : Number( $elems.css('opacity') ) | |

|  |  |
| --- | --- |
| 017 | }, |
| 018 | init        = function() { | |

|  |  |
| --- | --- |
| 019 |  |
| 020 | // minScale устанавливается в CSS | |

|  |  |  |
| --- | --- | --- |
| 021 | settings.minScale = \_getScaleVal() || 1; | |
| 022 | // Предварительная загрузка миниатюр |

|  |  |  |
| --- | --- | --- |
| 023 | \_loadImages( function() { | |
| 024 |  |

|  |  |
| --- | --- |
| 025 | \_calcDescrp(); |
| 026 | \_initEvents(); |

|  |  |
| --- | --- |
| 027 |  |
| 028 | }); | |

|  |  |
| --- | --- |
| 029 |  |
| 030 | }, | |

|  |  |  |
| --- | --- | --- |
| 031 | // Получаем значение масштаба из CSS с помощью JavaScript: | |
| 032 | \_getScaleVal= function() { |

|  |  |
| --- | --- |
| 033 |  |
| 034 | var st = window.getComputedStyle($elems.get(0), null), | |

|  |  |  |
| --- | --- | --- |
| 035 | tr = st.getPropertyValue("-webkit-transform") || | |
| 036 | st.getPropertyValue("-moz-transform") || |

|  |  |  |
| --- | --- | --- |
| 037 | st.getPropertyValue("-ms-transform") || | |
| 038 | st.getPropertyValue("-o-transform") || |

|  |  |  |
| --- | --- | --- |
| 039 | st.getPropertyValue("transform") || | |
| 040 | "fail..."; |

|  |  |
| --- | --- |
| 041 |  |
| 042 | if( tr !== 'none' ) { | |

|  |  |
| --- | --- |
| 043 |  |
| 044 | var values = tr.split('(')[1].split(')')[0].split(','), | |

|  |  |
| --- | --- |
| 045 | a = values[0], |
| 046 | b = values[1], |

|  |  |
| --- | --- |
| 047 | c = values[2], |
| 048 | d = values[3]; |

|  |  |
| --- | --- |
| 049 |  |
| 050 | return Math.sqrt( a \* a + b \* b ); | |

|  |  |
| --- | --- |
| 051 |  |
| 052 | } | |

|  |  |  |
| --- | --- | --- |
| 053 |  | |
| 054 | }, |

|  |  |  |
| --- | --- | --- |
| 055 | // Вычисляем значения для контейнера описания | |
| 056 | // на основе установочных переменных |

|  |  |  |
| --- | --- | --- |
| 057 | \_calcDescrp = function() { | |
| 058 |  |

|  |  |  |
| --- | --- | --- |
| 059 | $descrp.each( function(i) { | |
| 060 |  |

|  |  |
| --- | --- |
| 061 | var $el     = $(this), |
| 062 | $img    = $el.prev(), | |

|  |  |
| --- | --- |
| 063 | img\_w   = $img.width(), |
| 064 | img\_h   = $img.height(), | |

|  |  |
| --- | --- |
| 065 | img\_n\_w = settings.maxScale \* img\_w, |
| 066 | img\_n\_h = settings.maxScale \* img\_h, |

|  |  |
| --- | --- |
| 067 | space\_t = ( img\_n\_h - img\_h ) / 2, |
| 068 | space\_l = ( img\_n\_w - img\_w ) / 2; |

|  |  |
| --- | --- |
| 069 |  |
| 070 | $el.data( 'space\_l', space\_l ).css({ | |

|  |  |  |
| --- | --- | --- |
| 071 | height  : settings.maxScale \* $el.height(), | |
| 072 | top     : -space\_t, |

|  |  |  |
| --- | --- | --- |
| 073 | left    : img\_n\_w - space\_l | |
| 074 | }); |

|  |  |
| --- | --- |
| 075 |  |
| 076 | }); | |

|  |  |
| --- | --- |
| 077 |  |
| 078 | }, | |

|  |  |  |
| --- | --- | --- |
| 079 | \_initEvents = function() { | |
| 080 |  |

|  |  |  |
| --- | --- | --- |
| 081 | $elems.on('proximity.Photo', { max: 80, throttle: 10, fireOutOfBounds :true }, function(event, proximity, distance) { | |
| 082 |  |

|  |  |
| --- | --- |
| 083 | var $el         = $(this), |
| 084 | $li         = $el.closest('li'), | |

|  |  |
| --- | --- |
| 085 | $desc       = $el.next(), |
| 086 | scaleVal    = proximity \* ( settings.maxScale - settings.minScale ) + settings.minScale, | |

|  |  |  |
| --- | --- | --- |
| 087 | scaleExp    = 'scale(' + scaleVal + ')'; | |
| 088 |  |

|  |  |  |
| --- | --- | --- |
| 089 | // Изменяем z-index элемента, как только будет достигнуто максимальное значение масштаба. | |
| 090 | // Также показываем контейнер описания |

|  |  |  |
| --- | --- | --- |
| 091 | if( scaleVal === settings.maxScale ) { | |
| 092 |  |

|  |  |  |
| --- | --- | --- |
| 093 | $li.css( 'z-index', 1000 ); | |
| 094 |  |

|  |  |  |
| --- | --- | --- |
| 095 | if( $desc.offset().left + $desc.width() > listL + listW ) { | |
| 096 |  |

|  |  |  |
| --- | --- | --- |
| 097 | $desc.css( 'left', -$desc.width() - $desc.data( 'space\_l' ) ); | |
| 098 |  |

|  |  |  |
| --- | --- | --- |
| 099 | } | |
| 100 |  |

|  |  |  |
| --- | --- | --- |
| 101 | $desc.fadeIn( 800 ); | |
| 102 |  |

|  |  |
| --- | --- |
| 103 | } |
| 104 | else { | |

|  |  |
| --- | --- |
| 105 |  |
| 106 | $li.css( 'z-index', 1 ); | |

|  |  |
| --- | --- |
| 107 |  |
| 108 | $desc.stop(true,true).hide(); | |

|  |  |
| --- | --- |
| 109 |  |
| 110 | } | |

|  |  |
| --- | --- |
| 111 |  |
| 112 | $el.css({ | |

|  |  |  |
| --- | --- | --- |
| 113 | '-webkit-transform' : scaleExp, | |
| 114 | '-moz-transform'    : scaleExp, |

|  |  |
| --- | --- |
| 115 | '-o-transform'      : scaleExp, |
| 116 | '-ms-transform'     : scaleExp, | |

|  |  |
| --- | --- |
| 117 | 'transform'         : scaleExp, |
| 118 | 'opacity'           : ( proximity \* ( settings.maxOpacity - settings.minOpacity ) + settings.minOpacity ) | |

|  |  |  |
| --- | --- | --- |
| 119 | }); | |
| 120 |  |

|  |  |  |
| --- | --- | --- |
| 121 | }); | |
| 122 |  |

|  |  |
| --- | --- |
| 123 | }, |
| 124 | \_loadImages = function( callback ) { | |

|  |  |
| --- | --- |
| 125 |  |
| 126 | var loaded  = 0, | |

|  |  |  |
| --- | --- | --- |
| 127 | total   = $elems.length; | |
| 128 |  |

|  |  |  |
| --- | --- | --- |
| 129 | $elems.each( function(i) { | |
| 130 |  |

|  |  |  |
| --- | --- | --- |
| 131 | var $el = $(this); | |
| 132 |  |

|  |  |  |
| --- | --- | --- |
| 133 | $('<img/>').load( function() { | |
| 134 |  |

|  |  |
| --- | --- |
| 135 | ++loaded; |
| 136 | if( loaded === total ) | |

|  |  |  |
| --- | --- | --- |
| 137 | callback.call(); | |
| 138 |  |

|  |  |  |
| --- | --- | --- |
| 139 | }).attr( 'src', $el.attr('src') ); | |
| 140 |  |

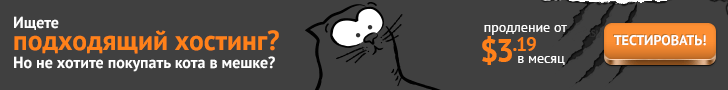
|  |  |  |
| --- | --- | --- |
| 141 | }); | |
| 142 |  |

|  |  |  |
| --- | --- | --- |
| 143 | }; | |
| 144 |  |

|  |  |
| --- | --- |
| 145 | return { |
| 146 | init    : init | |

|  |  |
| --- | --- |
| 147 | }; |

Данный урок подготовлен для вас командой сайта [ruseller.com](http://ruseller.com/)  
Источник урока: [tympanus.net/codrops/2012/01/04/thumbnail-proximity-effect/](http://ruseller.com/tympanus.net/codrops/2012/01/04/thumbnail-proximity-effect/)  
Перевел: Сергей Фастунов  
Урок создан: 2 Февраля 2012  
Просмотров: 43935  
[Правила перепечатки](http://ruseller.com/page.php?doc=reprint)

[](https://hostiq.com.ua/virtual-hosting/?utm_source=ruseller.com&utm_medium=sponsored&utm_campaign=ruseller-banner-flash)

## 5 последних уроков рубрики "jQuery"

* http://ruseller.com/lessons/les2788/thumbnail.png

#### [Вычисление позиции курсора мыши](http://ruseller.com/lessons.php?rub=32&id=2788)

Данный сниппет поможет вам определить координаты пользовательского курсора практически во всех браузерах.

* http://ruseller.com/lessons/les2787/thumbnail.png

#### [Отключение прокрутки на Google картах](http://ruseller.com/lessons.php?rub=32&id=2787)

Скорее всего каждый из нас хоть один раз вставлял Google карты на свои страницы. Тогда вы не могли не столкнуться с проблемой: во время прокрутки страницы, если курсор попадал на карту Google, то вместо смещения страницы, начинала масштабироваться карта. Мы ответим на вопрос о том, как отключить прокрутку в Google картах.+

* http://ruseller.com/lessons/les2781/thumbnail.jpeg

#### [16 jQuery плагинов для создания “каруселей”](http://ruseller.com/lessons.php?rub=32&id=2781)

Данные jQuery плагины позволят быстро и безболезненно создать слайдер контента.

* http://ruseller.com/lessons/les2780/thumbnail.jpg

#### [Извлечение GET параметров через JavaScript](http://ruseller.com/lessons.php?rub=32&id=2780)

Параметры в URL могут содержать кучу полезной информации: данные о товаре, пользовательские настройки, идентификатор партнёра. В этой статье посмотрим как через JS можно извлечь данную информацию.

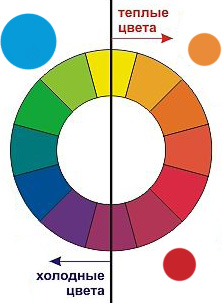
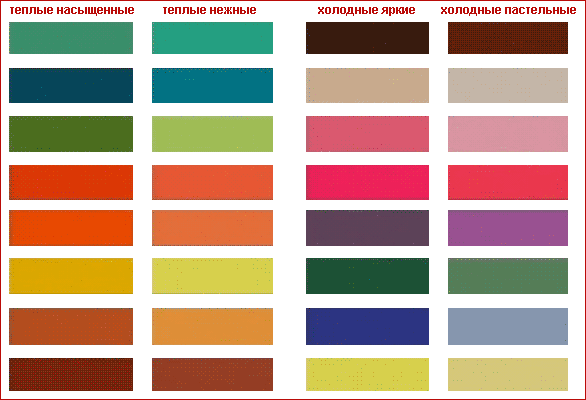
* http://ruseller.com/lessons/les2777/thumbnail.jpg

#### [Удаление элементов через JavaScript](http://ruseller.com/lessons.php?rub=32&id=2777)

Существует несколько методов удаления элементов через JavaScript: традиционный, перебирая элементы и новый - прямой метод DOM4.

**Приближающие и удаляющие цвета.**  
  
Продолжаем говорить о цвете и сегодня наша тема – действие цветов в пространстве.  Если говорить кратко, то

**Теплые  цвета приближают – холодные отдаляют.**

Теперь определим, что такое **теплые и холодные** цвета.  Если взять цветовой круг и разделить его на 2 части, то с одной стороны будут холодные цвета, с другой – теплые.  
[](http://artemari.livejournal.com/pics/catalog/363/14839)[](http://artemari.livejournal.com/pics/catalog/363/15001)  
  
Таким образом, если поклеить стену обоями теплых оттенков, то  она визуально будет казаться ближе. Если поклеить обоями холодных оттенков, например,  синим, или зеленым, то она будет казаться дальше.  Визуально приближают не только теплые цвета, но и темные.  Соответственно, светлые действуют, как холодные на удаление.  Соответственно, если представить комнату, одна из стен которой поклеена обоями темных оттенков, а остальные светлых, то одна стена будет казаться ближе.   
Существуют ситуации, при которых схема теплые-холодные меняется.  Это происходит при условии, когда  холодные оттенки становятся более яркими, а теплые более бледными.  Например, стена теплого розового оттенка будет казаться дальше, если с ней соседствуют ярко-синие стены.  Т.е при выборе оттенка необходимо учитывать не только его принадлежность к холодному или теплому, но и его цветовую силу.   
  
**Влияние на цвет источников освещения.**  
Существует 2 основных источника света: дневной (естественный) и искусственный.

В прямом солнечном свете предмет обычно освещен равномерно. Основной составляющий цвет солнечного света обычно белый, поэтому цвета предметов слегка «выбеляются»

Искусственный свет можно разделить на два вида: свет от ламп накаливания и свет от флуоресцентных ламп.  Лампы накаливания излучают свет, который тяготеет к желто-красному спектру. Поверхность, освещенная таким светом, приобретет желто-красный оттенок. Флуоресцентное освещение  (т.н лампы дневного света) дают синий оттенок, поэтому человек, освещенный таким светом кажется бледным. Поверхность, освещенная таким светом изменяет свой цвет в сторону синего спектра.  
  
**Что следует запомнить:**

1. **Теплые и яркие цвета приближаются.**
2. **Холодные и спокойные цвета остаются на своем месте или удаляются.**
3. **Темные цвета делают яркие предметы еще ярче.**
4. **Предметы холодного тона хорошо видны на теплом фоне.**
5. **Любой цвет изменяется под действием света.**
6. **Любой цвет меняется в присутствии другого цвета.**