**jQuery – это JavaScript-библиотека, фокусирующаяся на взаимодействии JavaScript, HTML и CSS.**

***Что умеет jQuery***

* Обращаться к любому элементу DOM (объектной модели документа) и не только обращаться, но и манипулировать ими.
* Работать с событиями.
* Легко осуществлять различные визуальные эффекты.
* Работать с AJAX (очень полезная технология, позволяющая общаться с сервером без перезагрузки страницы, но пока мы ее трогать не будем).
* Имеет огромное количество [JavaScript](http://www.site-do.ru/js/js.php) плагинов, предназначенных для создания элементов пользовательских интерфейсов.

***Как это работает***

Сначала надо скачать саму библиотеку [на сайте разработчиков](http://jquery.com/) или [с нашего сайта](http://site-do.ru/materials_for_lessons/jquery-1.2.6.zip), при необходимости разархивировать и перенести ее (библиотеку) в ту же папку, где лежат наши html-страницы (это необязательно, но адреса для всех последующих примеров будут написаны, исходя из такой структуры).

Теперь нам надо подключить jQuery к html-странице. Для этого, как вы помните, в html существует тег <script>, который и отвечает за подключение внешних файлов скриптов. Добавим этот тег в html-страницу (так же мы подключали страницу script.js с функциями js):

<html>

<head>

<title>jQuery</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

</body>

</html>

\* Обратите внимание на имя файла. Здесь используется библиотека jquery-1.2.6, но Вы можете скачать более свежую версию, а потому цифры у Вас могут отличаться.\*

Итак, у нас в папке лежат html-страница, страница со стилями (style.css), страница с js-функциями (script.js) и [библиотека jQuery](http://www.site-do.ru/js/jquery1.php) (jquery-1.2.6.js). Давайте сделаем вот такой пример (пощелкайте по кнопкам):

**Эффекты jQuery**

Эффект Show() Эффект SlideDown() Эффект Animate()

Давайте посмотрим, как это сделано. На html-странице у нас должно быть три прямоугольника (div-ы) и три кнопки:

<html>

<head>

<title>jQuery</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<h2>Эффекты jQuery</h2>

<div id="kv1" >1</div>

<div id="kv2" >2</div>

<div id="kv3" >3</div>

<button onclick="addEffect1()">Эффект Show()</button>

<button onclick="addEffect2()">Эффект SlideDown()</button>

<button onclick="addEffect3()">Эффект Animate()</button>

</body>

</html>

Оформим внешний вид и сделаем прямоугольники невидимыми с помощью [CSS](http://www.site-do.ru/css/). Код style.css:

#kv1, #kv2, #kv3{

width:80px;

height:100px;

float:left;

background:red;

margin:5px;

color:white;

display:none;

}

Теперь самое интересное, на странице script.js пишем код функций, которые будут срабатывать при нажатии на кнопки:

function addEffect1(){

$("#kv1:hidden").show();

}

function addEffect2(){

$("#kv2:hidden").slideDown("slow");

}

function addEffect3(){

$("#kv3:hidden").show().animate( {

fontSize:"36px"

} , 3000 );

}

Что же делают эти функции?

Функция addEffect1() видит $ (знак доллара) и понимает, что это jQuery, затем она видит ("#kv1:hidden") и понимает, что ей нужно найти элемент с id="kv1", обладающий свойством hidden (невидимый). Далее она видит .show() и понимает, что найденный элемент надо сделать видимым.

Функция addEffect2() видит $ (знак доллара) и понимает, что это jQuery, затем она видит ("#kv2:hidden") и понимает, что ей нужно найти элемент с id="kv2", обладающий свойством hidden (невидимый). Далее она видит .slideDown("slow") и понимает, что найденный элемент надо медленно ("slow") отобразить сверху-вниз (slideDown).

Функция addEffect3() видит $ (знак доллара) и понимает, что это jQuery, затем она видит ("#kv3:hidden") и понимает, что ей нужно найти элемент с id="kv3", обладающий свойством hidden (невидимый). Далее она видит .show() и понимает, что найденный элемент надо сделать видимым. Затем она видит .animate( {fontSize:"36px"} , 3000 ) и понимает, что размер шрифта нужно за 3 секунды (3000) увеличить до 36 пикселей (fontSize:"36px")

Синтаксис оператора jQuery можно представить следующим образом:

http://www.site-do.ru/images/jquery1.gif

где селектор - элемент или элементы, с которыми мы будем что-либо делать.

действие - что именно мы будем делать с выбранными элементами. Мы можем добавить какой-либо эффект, css-стиль, изменить html-код и т.д. Здесь же могут быть указаны какие-либо события.

свойства действия - если они предусмотрены действием.

**JQuery селекторы**

Селекторы - это элементы, с которыми мы хотим что-либо сделать. Давайте посмотрим, как их можно задавать:

#id - выбирает единственный элемент с переданным идентификатором (id).

Пример:

* $("#lok").css("border", "1px solid red");

Данный оператор выберет элемент с id="lok" и добавит ему css-стиль, в данном случае рамку, шириной в 1 пиксел, сплошную и красную. (О способах задания css-стилей будет говориться в отдельном уроке).

.class - выбирает все элементы с переданным классом.  
Пример:

* $(".lok").css("border", "1px solid red");

Данный оператор выберет все элементы с class="lok" и добавит им [css-стиль](http://www.site-do.ru/css/), в данном случае рамку, шириной в 1 пиксел, сплошную и красную.

elements - выбирает все элементы с выбранным именем.

Пример:

* $("div").css("border", "1px solid red");

Данный оператор выберет все div-ы и обведет их в рамки.

\* - выбирает все элементы, включая head и body.

Пример:

* $("\*").css("border", "1px solid red");

Данный оператор добавит рамку ко всем элементам.

$("\*", document.body).css("border", "1px solid red");

Данный оператор добавит рамку ко всем элементам в тегах body.

selector1, ..., selectorN - выбирает все элементы, переданные в селекторах.  
Пример:

* $("div, span, p.lok").css("border", "1px solid red");

Данный оператор добавит рамку ко всем div-ам, span-ам, и абзацам, имеющим класс lok. Таким образом, через зяпятую можно указать любые селекторы.

Давайте попрактикуемся и сделаем вот что:

Селекторы jQuery

Текст первого абзаца

Текст второго абзаца

Текст третьего абзаца

Сделать первый div красным

Обвести средний абзац синей рамкой

Сделать шрифт в первом и третьем div-ах жирным

Код html-страницы прост:

<html>

<head>

<title>jQuery</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<h2>Селекторы jQuery</h2>

<div id="st1" ><p>Текст первого абзаца</p></div>

<div id="st2" ><p class="withBorder">Текст второго абзаца</p></div>

<div id="st3" ><p>Текст третьего абзаца</p></div>

<div style="clear: both"></div>

<button onclick="addStyle1()">Сделать первый div красным</button><br><br>

<button onclick="addStyle2()">Обвести средний абзац синей рамкой</button><br><br>

<button onclick="addStyle3()">Сделать шрифт в первом и третьем div-ах жирным</button>

</body>

</html>

Внешний вид зададим с помощью таблицы стилей (style.css):

#st1, #st2, #st3{

float:left;

background:yellow;

margin:5px;

padding:5px;

}

Ну и на странице script.js напишем сами функции:

function addStyle1(){

$("#st1").css("background", "red");

}

function addStyle2(){

$("p.withBorder").css("border", "1px solid blue");

}

function addStyle3(){

$("#st1, #st3").css("fontWeight", "bold");

}

**Методы для работы с CSS-стилями**

В jQuery имеется три категории методов: одни манипулируют с элементами, подходящими по шаблону; вторые возвращают значения элемента, а третьи изменяют сами элементы.

Сегодня рассмотрим методы, которые используются для CSS-стилей.

Итак, чтобы добавить какому-либо элементу стиль, необходимо воспользоваться следующим методом:

.css(name,value)

Пример:

$("div").css("border", "1px solid blue");

Данная инструкция обведет div синей рамкой.

В качестве параметров здесь используются названия и значения, применимые в [CSS](http://www.site-do.ru/css/): [background](http://www.site-do.ru/css/css5.php), [border](http://www.site-do.ru/css/css9.php), font-style, color и т.д.

Если необходимо задать для элемента несколько CSS-правил, то лучше использовать следующую конструкцию:

.css({properties})

Пример:

$("div").css({

border:"1px solid blue",

fontWeight:"bolder",

backgroundColor:"red"

});

Данная инструкция обведет div синей рамкой, сделает фон красным, а текст - жирным.

Обратите внимание, что для сложносоставных свойств CSS вроде font-weight и background-color используются иэ эквиваленты из JS: fontWeight, backgroundColor и т.д.

Перечислим другие методы для работы со стилями:

.addClass(class)

Пример:

* $("p:last").addClass("main");

Данная инструкция добавит класс main к последнему элементу параграфа.

.removeClass(class)

Пример:

* $("p:even").removeClass("main");

Данная инструкция удалит класс main из всех четных параграфов.

.toggleClass(class)

Пример:

* $("p").toggleClass("main");

Данная инструкция удалит класс main из всех параграфов, если он присутствует. И добавит этот класс, если он отсутствует.

.offset()

Пример:

* var offDiv=$("div").offset();

Данная инструкция позволяет получить [отступы слева и сверху](http://www.site-do.ru/css/css9.php) для элемента. Чтобы получить значения конкретного свойства, нужно использовать следующие свойства: offset.left для отступа слева и offset.top - для отступа сверху.

.height(value)

Пример:

* $("div").height();
* $("div").height(200);

Данная инструкция позволяет получить (первая строка) и задать (вторая строка) высоту элемента.

.width(value)

Пример:

* $("div").width();
* $("div").width(200);

Данная инструкция позволяет получить (первая строка) и задать (вторая строка) ширину элемента.

Еще пример:

var widDiv=$("div").width();

$("div.fir").width(300);

Первая строка запишет в переменную widDiv значение ширины первого div-а. Вторая инструкция задаст div-ам класса fir ширину в 300 пикселов.

Это очень интересная особенность методов jQuery: они используются, как для задания параметров (когда принимаются 2 аргумента), так и для получения значений этих параметров (если передается один аргумент).

Давайте сделаем меню с вкладками:

jQuery - CSS

Вкладка 1

Вкладка 2

Вкладка 3

Вкладка 4

Итак, html-код страницы следующий:

<html>

<head>

<title>jQuery</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<h2>jQuery - CSS</h2>

<div id="vkladki">

<div class="vkl" id="vkladka1" onClick="chang('#vkladka1');">Вкладка 1</div>

<div class="vkl" id="vkladka2" onClick="chang('#vkladka2');">Вкладка 2</div>

<div class="vkl" id="vkladka3" onClick="chang('#vkladka3');">Вкладка 3</div>

<div class="vkl" id="vkladka4" onClick="chang('#vkladka4');">Вкладка 4</div>

</div>

<div style="clear: both"></div>

<div id="content"></div>

<div style="clear: both"></div>

</body>

</html>

Как видно, все вкладки имеют единый стиль, определяемый классом vkl, а выделенная вкладка имеет класс selected. Внешний вид пропишем на странице style.css

.vkl{

float:left;

width:140px;

height:28px;

background:#E5BA83;

color:white;

font-size:20px;

border-right:1px solid white;

padding-left:10px;

}

#content{

width:603px;

height:100px;

background:#CCA675;

}

.selected{

background:#CCA675;

}

Осталось самое интересное, а именно функция chang():

function chang(s){

$('.selected').removeClass('selected');

$(s).addClass('selected');

}

Итак, как же работает эта функция? Сначала ищется элемент, который имеет класс selected, и этот класс у него убирается. Затем (вторая строка) выделенному элементу (он находится по id, переданному в функцию в качестве параметра s) добавляется класс selected.

**Методы для работы с html**

Рассмотрим основные [методы работы с html](http://www.site-do.ru/js/jquery5.php), а затем уже методы для манипуляции с этими элементами. Итак, основные методы:

html(val) - добавляет html-код в выбранные элементы.

Пример:

* $("div.sp").html("<span>Привет</span>");

Данная инструкция добавит во все div-ы с классом sp, span-ы с текстом "Привет".

text() / val() - возвращает текстовое содержимое элемента / значение элемента.

Пример:

* $("p").text();

Данная инструкция вернет текст из параграфа.

text(val) / val(val) - установит текст для элемента / значение для элемента.

Пример:

* $("p").text("Привет!");

Данная инструкция в параграфе напишет слово "Привет!".

Давайте для примера, сделаем так, чтобы при нажатии по кнопке "Повторить", текст из первого абзаца дублировался во второй абзац.

Текст первого абзаца!

Html-код:

<html>

<head>

<title>jQuery html</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<p id="p1">Текст первого абзаца!</p>

<p id="p2"></p>

<input type="button" value="Повторить!" onclick="addHtml();">

</body>

</html>

Фукция будет выглядеть так:

function addHtml(){

var tp1=$("#p1").text();

$("#p2").text(tp1);

}

Т.е. сначала в переменную tp1 мы запишем текстовое содержимое первого абзаца (первая строка), а затем передадим его во второй абзац (вторая строка).

Теперь давайте рассмотрим методы, которые позволяют манипулировать вставляемыми элементами.

append(content) - добавляет content внутрь всех выбранных элементов ПОСЛЕ существующего контента.

Пример:

* $("p").append("<b>Привет</b>");

Данная инструкция добавит в конец параграфа текст "Привет" жирным шрифтом.

appendTo(content) - обратная операция, добавляет В content все выбранные элементы.

Пример:

* $("p").appendTo("#main");

Данная инструкция добавит параграф в конец элемента с идентификатором "main".

Иными словами:

$(A).append(B) - добавит В в конец А.

$(A).appendTo(B) - добавит А в конец В.

prepend(content) - добавляет content внутрь всех выбранных элементов ДО существующего контента.

Пример:

* $("p").prepend("<b>Привет</b>");

Данная инструкция вставит в начало параграфа текст "Привет" жирным шрифтом.

prependTo(content) - обратная операция, добавляет В content все выбранные элементы.

Пример:

* $("p").prependTo("#main");

Данная инструкция вставит параграф в начало элемента с идентификатором "main".

Иными словами:

$(A).prepend(B) - добавит В в начало А.

$(A).prependTo(B) - добавит А в начало В.

after(content) - добавляет content ПОСЛЕ всех выбранных элементов (заметьте ПОСЛЕ элемента, а не в конец элемента, как в случае с append.

Пример:

* $("p").after("<b>Привет</b>");

Данная инструкция вставит после параграфа текст "Привет" жирным шрифтом.

befor(content) - добавляет content ДО всех выбранных элементов (заметьте ДО элемента, а не в начало элемента, как в случае с prepend.

Пример:

* $("p").befor("<b>Привет</b>");

Данная инструкция вставит перед параграфом текст "Привет" жирным шрифтом.

insertAfter(content) - вставляет все выбранные элементы ПОСЛЕ content-а .

Пример:

* $("p").insertAfter("#main");

Данная инструкция вставит параграф после элемента с идентификатором "main".

Иными словами:

$(A).after(B) - вставит В после А.

$(A).insertAfter(B) - вставит А после В.

insertBefore(content) - вставляет все выбранные элементы ПЕРЕД content-ом .

Пример:

* $("p").insertBefore("#main");

Данная инструкция вставит параграф перед элементом с идентификатором "main".

Иными словами:

$(A).before(B) - вставит В перед А.

$(A).insertBefore(B) - вставит А перед В.

Рассмотрим пример. Давайте сделаем так, чтобы при нажатии на кнопку "Добавить" в красном прямоугольнике появлялись желтые квадратики (один клик - один квадратик).

Код html-страницы состоит из одного div-а и одной кнопки:

<html>

<head>

<title>jQuery html</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="red\_sk"></div>

<input type="button" value="Добавить!" onclick="addYellowSquare();">

</body>

</html>

На странице style.css как всегда определяем стили:

#red\_sk{

width:700px;

height:100px;

background:red;

}

.yellow\_square{

float:left;

width:80px;

height:80px;

background:yellow;

margin:5px;

}

Ну а код самой функции совсем короткий:

function addYellowSquare(){

$("#red\_sk").prepend("<div class='yellow\_square'></div>");

}

Вот так все просто. Но продолжим рассматривать сами методы:

wrap(html) - оборачивает каждый элемент в элемент обертку.

Пример:

* $("p").wrap("<span></span>");

Данная инструкция обернет каждый параграф span-ом.

wrapAll(html) - оборачивает все элементы в ОДИН элемент обертку.

Пример:

* $("p").wrapAll("<span></span>");

Данная инструкция обернет все параграфы одним span-ом.

wrapInner(html) - оборачивает внутренне содержание каждого элемента.

Пример:

* $("p").wrapInner("<em></em>");

Данная инструкция выделит содержимое каждого параграфа курсивом.

replaceWith(content) - замещает одни элементы другими.

Пример:

* $("span").replaceWith("<div></div>");

Данная инструкция заменит все span-ы div-ами.

replaceAll(selector) - обратная операция, т.е. все selector-ы будут поменяны на элементы.

Пример:

* $("span").replaceAll("<div></div>");

Данная инструкция заменит все div-ы span-ами.

empty() - удаляет из элемента все узлы-потомки, включая текст.

Пример:

* $("span").empty();

Данная инструкция удалит все содержимое span-ов.

remove() - удаляет сами элементы.

Пример:

* $("span").remove();

Данная инструкция удалит все span-ы.

clone() - клонирует элементы.

Пример:

* $("b").clone().append("div");

Данная инструкция возьмет элементы, выделенные жирным и добавит их в div.

**Визуальные эффекты**

Методы видимости - hide(), show() и toggle()

Эти методы отвечают за видимость элементов, работают по принципу скрыть - показать:

**hide() - скрыть,**

**show() - показать,**

**toggle() - показать, если скрыты и скрыть, если видимы.**

Пример:

Html-код:

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex1"></div>

<input type="button" value="Скрыть" onclick="hideDiv();">

<input type="button" value="Показать" onclick="showDiv();">

</body>

</html>

Код функций (на странице script.js):

function hideDiv(){

$('#les8\_ex1').hide();

}

function showDiv(){

$('#les8\_ex1').show();

}

Ну и стили на style.css в принципе на ваше усмотрение:

#les8\_ex1{

width:400px;

height:100px;

background:#666699;

border:1px solid #333366;

margin-bottom:20px;

}

Страница style.css здесь приведена не случайно. Все дело в том, что скрыть элемент можно и при помощи таблицы стилей, а не методом hide().

Например, если бы мы хотели, чтобы при загрузке страницы наш div был скрыт, то мы могли бы это сделать двумя способами:

Первый, использовать метод hide():

$(document).ready(init);

function init(){

$('#les8\_ex1').hide();

}

function hideDiv(){

$('#les8\_ex1').hide();

}

function showDiv(){

$('#les8\_ex1').show();

}

А второй - использовать свойство [display](http://www.site-do.ru/css/css12.php) CSS:

#les8\_ex1{

width:400px;

height:100px;

background:#666699;

border:1px solid #333366;

margin-bottom:20px;

display:none;

}

Результат будет одинаков.

Эти же методы гораздо интереснее использовать с анимацией:

**hide(speed, callback) - скрыть,**

**show(speed, callback) - показать,**

**toggle(speed, callback) - переключить (показать, если скрыты и наоборот), где:**

speed - скорость изменения высоты, ширины или свойства opacity (прозрачность) элемента. Может принимать три значения: slow (медленно), normal (нормально) или fast (быстро), а также значение в миллисекундах.

callback - функция, которая будет выполняться после завершения анимации. Ее присутствие вовсе необязательно.

Посмотрим тот же пример, только скрывать и отображать div будем медленно (сравните с первым примером) и, используя метод toggle().

Итак, html-код:

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex2"></div>

<input type="button" id="but1" value="Скрыть/Отобразить" onclick="hideShowDiv();">

</body>

</html>

Код функции:

function hideShowDiv(){

$('#les8\_ex2').toggle('slow');

}

**Методы свертывания - slideDown(), slideUp() и slideToggle()**

Эти методы также отвечают за видимость элементов, но работают по принципу свернуть элемент плавно снизу-вверх - развернуть элемент плавно свеху-вниз:

slideUp() - свернуть,

slideDown() - развернуть,

toggle() - развернуть, если скрыт и свернуть, если видим.

Пример:

Html-код:

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex3"></div>

<input type="button" value="Свернуть" onclick="slideUpDiv();">

<input type="button" value="Развернуть" onclick="slideDownDiv();">

</body>

</html>

Код функций (на странице script.js):

function slideUpDiv(){

$('#les8\_ex3').slideUp();

}

function slideDownDiv(){

$('#les8\_ex3').slideDown();

}

[Стили](http://www.site-do.ru/css/css7.php) на style.css:

#les8\_ex3{

width:100px;

height:200px;

background:#666699;

border:1px solid #333366;

margin-bottom:20px;

}

Эти методы также можно использовать с анимацией:

slideUp(speed, callback) - свернуть,

slideDown(speed, callback) - развернуть,

slideToggle(speed, callback) - переключить (развернуть, если скрыты и наоборот), где:

speed - скорость изменения высоты элемента. Может принимать три значения: slow (медленно), normal (нормально) или fast (быстро), а также значение в миллисекундах.

callback - функция, которая будет выполняться после завершения анимации. Ее присутствие необязательно.

Посмотрим тот же пример, только скрывать и отображать div будем за 7 секунд и, используя метод toggle().

Html-код:

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex4"></div>

<input type="button" value="Свернуть/Развернуть" onclick="slideToggleDiv();">

</body>

</html>

Код функций (на странице script.js):

function slideToggleDiv(){

$('#les8\_ex4').slideToggle(7000);

}

**Методы исчезновения - fadeTo(), fadeOut() и fadeIn()**

fadeTo(speed, opacity, callback) - уменьшает свойство opacity (прозрачность) к заданному значению,

fadeOut(speed, callback) - уменьшает свойство opacity (прозрачность),

fadeIn(speed, callback) - увеличивает свойство opacity (прозрачность),

где:

speed - скорость изменения прозрачности элемента. Может принимать три значения: slow (медленно), normal (нормально) или fast (быстро), а также значение в миллисекундах.

opacity - значение прозрачности, до которого оно будет уменьшено (число от 0 до 1).

callback - функция, которая будет выполняться после завершения анимации. Ее присутствие необязательно.

**Например:**

[Html-код:](http://www.site-do.ru/html/)

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex5"></div>

<input type="button" value="Исчезает (fadeOut)" onclick="fadeOutDiv();">

<input type="button" value="Появляется (fadeIn)" onclick="fadeInDiv();">

<div id="les8\_ex6"></div>

<input type="button" value="Бледнеет (fadeTo)" onclick="fadeToDiv();">

</body>

</html>

Код функций (на странице script.js):

function fadeOutDiv(){

$('#les8\_ex5').fadeOut(5000);

}

function fadeInDiv(){

$('#les8\_ex5').fadeIn(5000);

}

function fadeToDiv(){

$('#les8\_ex6').fadeTo(5000, 0.5);

}

**Методы анимации - animate() и stop()**

animate(params, options) - анимирует стилевые свойства, где:

params - атрибуты css, которые хотим анимировать ("width", "top", "border"...).

options - свойства анимации (в том числе скорость).

stop() - останавливает анимацию.

**Например:**

Html-код:

<html>

<head>

<title>jQuery эффекты</title>

<link rel="stylesheet" type="text/css" href="style.css">

<script type="text/javascript" src="jquery-1.2.6.js"></script>

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

</head>

<body>

<div id="les8\_ex7"></div><div id="les8\_ex8"></div>

<input type="button" value="Запустить анимацию" onclick="animateDiv();">

<input type="button" value="Остановить анимацию" onclick="stopDiv();">

</body>

</html>

Стили (на странице style.css):

#les8\_ex7{

width:100px;

height:100px;

background:#666699;

}

#les8\_ex8{

width:400px;

height:200px;

background:#666699;

}

Код функций (на странице script.js):

function animateDiv(){

$('#les8\_ex7').animate({

width:"400px",

height:"200px"

}, 3000 );

$('#les8\_ex8').animate({

width:"100px",

height:"100px"

}, 3000 );

}

function stopDiv(){

$('#les8\_ex7').stop();

$('#les8\_ex8').stop();

}

Несколько нюансов:

- свойства должны быть обозначены без пробелов, последующее слово с большой буквы, т.е. "borderWidth" вместо "border-width",

- поддерживаются только те свойства, значения которых выражаются числами.

- также в качестве значений свойств могут быть использованы значения "hide", "show" и "toggle".

Например, следующая функция:

function animateDiv(){

$('#les8\_ex8').animate({

"height": "toggle"

}, 1000 );

}

будет при каждом обращении менять высоту от 0 до заданной в стилях и обратно: