Селекторы jQuery

С помощью селекторов jQuery Вы можете выбрать любой элемент на странице.

Все возможные варианты использования селекторов перечислены в таблице ниже:

Пример

Описание

$('*')

Будут выбраны все элементы присутствующие на странице.

$("#el1")

Будет выбран элемент с id=el1.

$(".el1")

Будут выбраны все элементы на странице с class=el1.

$("div")

Будут выбраны все элементы div на странице.

$("div, #el1, .el1")

Будут выбраны все элементы div, элемент с id="el1" и все элементы с class="e

$("div.el1")

Будут выбраны все элементы div на странице атрибут class которых равен el1.

$("p > div")

Будут выбраны все элементы потомки div родительского элемента p.

$("[src]")

Будут выбраны все элементы на странице имеющие атрибут src.

$("[src='wisdomweb.gif']")

Будут выбраны все элементы на странице со значениями атрибута src="wisdomweb.gif&c

$("[src!='wisdomweb.gif']")

Будут выбраны все элементы на странице со значениями атрибута src не равными "wisd

$("[src^='wisdomweb']")

Будут выбраны все элементы на странице со значениями атрибута src начинающимися на wis

$("[src$='.gif']")

Будут выбраны все элементы на странице со значениями атрибута src заканчивающимися на

$("[src*='wisdomweb']")

Будут выбраны все элементы на странице со значениями атрибута src содержащими wisdomw

$(":input")

Будут выбраны все элементы input на странице.

$(":button")

Будут выбраны все элементы input на странице с атрибутом type=button.

$(":text")

Будут выбраны все элементы input на странице с атрибутом type=text.

$(":password")

Будут выбраны все элементы input на странице с атрибутом type=password.

$(":radio")

Будут выбраны все элементы input на странице с атрибутом type=radio.

$(":checkbox")

Будут выбраны все элементы input на странице с атрибутом type=checkbox.

$(":reset")

Будут выбраны все элементы input на странице с атрибутом type=reset.

$(":image")

Будут выбраны все элементы input на странице с атрибутом type=image.

$(":file")

Будут выбраны все элементы input на странице с атрибутом type=file.

$("div:first")

Будет выбран первый div элемент на странице.

$("div:last")

Будет выбран последний div элемент на странице.

$("li:even")

Будут выбраны все элементы списка с четными индексами. Так как нумерация индексов элем

$("li:odd")

Будет выбраны все элементы с нечетными индексами. Так как нумерация индексов элементо

$("li:eq(3)")

Будет выбран 4 элемент списка (нумерация элементов в списке начинается с нуля).

$("li:gt(1)")

Будет выбраны все элементы списка индекс которых больше 1 (т.е. все элементы списка нач

$("li:lt(2)")

Будет выбраны все элементы списка индекс которых меньше 2 (т.е. первые 2 элемента списк

$(":header")

Будет выбраны все элементы на странице являющиеся заголовками (т.е. элементы h1, h2, h5

$(":animated")

Будет выбраны все анимированные элементы, которые находятся на странице.

$(":contains('wisdomweb')")

Будет выбраны все элементы содержащие строку wisdomweb.

$(":empty")

Будет выбраны все элементы не имеющие узлов потомков.

$(":hidden")

Будет выбраны все скрытые элементы на странице.

$(":visible")

Будет выбраны все видимые элементы находящиеся на странице.

Пример:

```
$(document).ready(function(){

//Установим размер шрифта всех абзацев равным 20  пикселям

  $(&amp;quot;p&amp;quot;).css(&amp;quot;fontSize&amp;quot;,&amp;quot;20px&amp;quot;);

//Изменим на зеленый цвет шрифта элемента с id=el2

  $(&amp;quot;#el2&amp;quot;).css(&amp;quot;color&amp;quot;,&amp;quot;green&amp;quot;);

//Изменим на красный цвет шрифта элемента с class=el3
```

```
   $(&quot;.el3&quot;).css(&quot;color&quot;,&quot;red&quot;);

//Сделаем жирным шрифт элементов с id=el2 и class=el3

   $(&quot;#el2,.el3&quot;).css(&quot;fontWeight&quot;,&quot;bold&quot;);

//Изменим на синий цвет текста кнопки

   $(&quot;:input&quot;).css(&quot;color&quot;,&quot;blue&quot;);

//Установим размер шрифта всех элементов имеющих  атрибут href равным 20 пикселям

   $(&quot;[href]&quot;).css(&quot;fontSize&quot;,&quot;20px&quot;);

//Изменим на зеленый цвет ссылки на www.wisdomweb.ru

   $(&quot;[href='http://www.wisdomweb.ru/']&quot;).css(&quot;color&quot;,&quot;green&quot;);



});
```

Предотвращение преждевременного выполнения кода

Из учебника по JavaScript Вы наверно помните, что выполнение кода до полной загрузки документа часто приводит к ошибкам.

Дело в том, что скрипт может обратится к еще не загруженному содержимому, а это всегда приводит к ошибкам или неожиданным результатам.

Для того, чтобы избежать этого мы часто помещали код в функцию, которая начинала выполнение только после полной загрузки документа.

В jQuery можно избавиться от преждевременного выполнения кода следующими способами:

Предотвращение преждевременного выполнения кода в jQuery:

<script type='text/javascript'> //Первый способ: $(document).ready(function(){ Код кото
рый
будет
выполнен
после
полной
загрузки
документа

});

//Второй способ:

$
().
ready

```
(
function
(){


Код
который
будет
выполнен
после
полной
загрузки
документа

});


//Третий способ:

$
(
function
(){


Код
который
будет
выполнен
после
полной
загрузки
документа

});

</script>
```

Существует еще один альтернативный способ также помогающий избежать преждевременное выполнение JavaScript и jQuery кода и позволяющий также ускорить загрузку страниц (*благодарим за напоминание о нем пользователя Ghringo Americano*).

Необходимо помещать код в самый конец тела документа (т.е. перед </body>) в данном

случае интерпретатор JavaScript встроенный в браузер начнет разбирать код только **по сле загрузки**
документа. В предыдущем же способе загрузка скриптов происходит
одновременно с загрузкой
документа, а выполняется этот код после загрузки документа.

<body> Содержимое тела документа Содержимое тела документа

<script
type
=
'text/javascript'
>

Код
который
будет
выполнен
после
полной
загрузки
документа

</script>

</body>

Цепочки команд в jQuery

Для того, чтобы сократить размер кода Вы можете соединять команды jQuery в цепочки.

Команды в цепочке будут выполняться поочередно слева направо.

<script type='text/javascript'> //Код без сокращения $("p").css("color" ,"green "); $("p"). css

```
(
&quot;font-size&quot;
,
&quot;30px&quot;
);

//
Сокращенный
код

$
(
&quot;p&quot;
).
css
(
&quot;color&quot;
,
&quot;green&quot;
).
css
(
&quot;font-size&quot;
,
&quot;30px&quot;
);

</script>
```

Обработчики событий jQuery

С помощью методов перечисленных в таблице ниже Вы можете создавать обработчики событий и привязывать их к элементам.

Синтаксис:

//Привяжем к кнопке обработчик события click

$(":button").click(function(){$("#par1").html("Новый текст")})

//Вызовем привязанный ранее обработчик

$(":button").click()

Обратите внимание: для того, чтобы узнать о желаемом методе или событие больше щелкните на его названии мышкой.

Метод

Описание

blur()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент пе

change()

Привязывает или вызывает функцию, код которой будет выполнен в случае изменения содер

click()

Привязывает или вызывает функцию, код которой выполнится, когда на выбранном элементе

dblclick()

Привязывает или вызывает функцию, код которой выполнится, когда на выбранном элементе

error()

Является устаревшим в jQuery 1.8. Привязывает или вызывает функцию, код которой выполнится если при

focus()

Привязывает или вызывает функцию, код которой будет выполнен, когда выбранный элемент

focusin()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент или

focusout()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент или

hover()

Привязывает одну или две функции к выбранному элементу. Код первой привязанной функции

keydown()

Привязывает или вызывает функцию, код которой выполнится, когда на клавиатуре будет на

keyup()

Привязывает или вызывает функцию, код которой выполнится, когда нажатая на клавиатуре к

load()

Является устаревшим в jQuery 1.8+ Привязывает или вызывает функцию, код которой будет выполнен посл

mousedown()

Привязывает или вызывает функцию, код которой будет выполнен после нажатия клавиши мы

mouseenter()

Привязывает или вызывает функцию, код которой будет выполнен, когда на выбранный элеме

mouseleave()

Привязывает или вызывает функцию, код которой будет выполнен, когда указатель мыши буд

mousemove()

Привязывает или вызывает функцию, код которой будет выполнен при передвижении указате

mouseout()

Привязывает или вызывает функцию, код которой будет выполнен, когда указатель мыши буд

mouseover()

Привязывает или вызывает функцию, код которой будет выполнен, когда на выбранный элеме

mouseup()

Привязывает или вызывает функцию, код которой будет выполнен, когда нажатая кнопка мыш

ready()

Привязывает или вызывает функцию, код которой будет выполнен, когда страница будет по

resize()

Привязывает или вызывает функцию, код которой будет выполнен при изменении размера окн

scroll()

Привязывает или вызывает функцию, код которой будет выполнен при прокрутке содержимог

select()

Привязывает или вызывает функцию, код которой будет выполнен при выделении текста вы

submit()

Привязывает или вызывает функцию, код которой будет выполнен при отправлении содержим

unload()

Является устаревшим в jQuery 1.8 Привязывает или вызывает функцию, код которой будет выполнен при в

Управление обработчиками событий

С помощью методов представленных в таблице ниже Вы сможете управлять обработчиками событий.

Метод

Описание

bind()

Привязывает к выбранному элементу один обработчик события или более.

delegate()

Добавляет один обработчик события или более к элементам потомкам выбранного элемента.

one()

Привязывает к выбранному элементу один обработчик события или более. Привязанные обра

toggle()

Позволяет привязать к выбранному элементу несколько обработчиков событий, между вызов

trigger()

Вызывает указанный обработчик события у выбранного элемента.

triggerHandler()

Вызывает указанный обработчик события у выбранного элемента.

unbind()

Удаляет у выбранных элементов обработчики событий, которые были привязаны с помощью

undelegate()

Удаляет у выбранных элементов обработчики событий, которые были привязаны с помощью

Объект event

Объект event хранит информации о произошедшем событии.

Объект event создается для каждого произошедшего события, но для того чтобы иметь возможность обратится к его свойствам и методам, его необходимо явно передать в обработчик события.

Синтаксис:

```
$(селектор).событие(function(event){

  //Затем в коде обработчика Вы можете обращаться к его свойствам и методам
```

следующим образом:

```
  event.data;

  event.preventDefault();

});
```

Обратите внимание: при передаче объекта event в обработчик Вы можете использовать любое имя.

Свойство	**Описание**
currentTarget	Содержит имя DOM элемента, в котором произошло событие.
data	Содержит дополнительные данные переданные обработчику события во время привязки его к

pageX

Содержит координаты указателя мыши по оси X во время вызова события.

pageY

Содержит координаты указателя мыши по оси Y во время вызова события.

result

Содержит последнее значение возвращенное вызванным ранее обработчиком события.

target

Содержит имя DOM элемента, который вызвал событие.

timeStamp

Содержит количество прошедших с 1 Января 1970 года миллисекунд до вызова данного собы

type

Содержит тип (название) произошедшего события.

which

Содержит код кнопки, которая была зажата во время вызова данного события.

Метод

Описание

isDefaultPrevented()

Позволяет узнать был ли вызван метод preventDefault() для данного элемента.

isImmediatePropagationStopped()

Позволяет узнать был ли вызван метод stopImmediatePropagation() для данного элемента.

isPropagationStopped()

Позволяет узнать был ли вызван метод stopPropagation() для данного элемента.

preventDefault()

Предотвращает выполнение стандартного действия элемента.

stopImmediatePropagation()

Запрещает вызов остальных обработчиков события привязанных к элементу.

stopPropagation()

Останавливает "всплытие" вызова события к родительским элементам.

Эффекты jQuery

С помощью jQuery методов **fadeOut()**, **fadeIn()** и **fadeTo()** Вы можете постепенно скрывать и отображать элементы анимировано.

Синтаксис:

//Позволяет постепенно скрыть выбранный элемент

$("селектор").fadeOut(скорость,функция);

//Позволяет постепенно отобразить выбранный элемент

$("селектор").fadeIn(скорость,функция);

//Позволяет постепенно скрыть/отобразить элемент до указанного значения прозрачности

$("селектор").fadeTo(скорость,прозрачность,функция);

Пример

$(document).ready(function(){

```
  $(&quot;#but1&quot;).click(function(){$(&quot;#par1&quot;).fadeOut(3000)});

  $(&quot;#but2&quot;).click(function(){$(&quot;#par1&quot;).fadeIn(3000)});

  $(&quot;#but3&quot;).click(function(){$(&quot;#par1&quot;).fadeTo(3000,0.3)});

  $(&quot;#but4&quot;).click(function(){$(&quot;#par1&quot;).fadeTo(3000,0.8)});

  $(&quot;#but5&quot;).click(function(){$(&quot;#par1&quot;).fadeOut(3000,function(){

    alert(&quot;Абзац был полностью скрыт.&quot;);

  });

  });



});
```

Попробовать в редакторе

С помощью jQuery методов **slideUp**, **slideDown** и **slideToggle** Вы можете плавно изменять высоту выбранных элементов.

Синтаксис:

//Позволяет изменяет высоту элемента до 0

$("селектор").slideUp(скорость,функция);

//Позволяет плавно возвратить элементу его изначальную высоту

$("селектор").slideDown(скорость,функция);

//При первом вызове будет действовать как slideUp, а при втором как slideDown

$("селектор").slideToggle(скорость,функция);

Пример

$(document).ready(function(){

```
$(&quot;#but1&quot;).click(function(){$(&quot;#square&quot;).slideUp(3000)});

$(&quot;#but2&quot;).click(function(){$(&quot;#square&quot;).slideDown(3000)});

$(&quot;#but3&quot;).click(function(){$(&quot;#square&quot;).slideToggle(3000)});

$(&quot;#but4&quot;).click(function(){$(&quot;#square&quot;).slideUp(3000,function(){

  alert(&quot;Текст был скрыт&quot;);

});

});



});
```

Попробовать в редакторе

С помощью метода slideToggle Вы можете создавать на Ваших страницах удобные выпадающие меню.

Пример

```
$(document).ready(function(){

  $(&quot;#menu&quot;).click(function(){$(&quot;#list&quot;).slideToggle(2000)});

  $(&quot;#menu&quot;).toggle(function(){

    $(&quot;#img&quot;).attr(&quot;src&quot;,&quot;menudown.gif&quot;)},function(){

    $(&quot;#img&quot;).attr(&quot;src&quot;,&quot;menuup.gif&quot;)

  });

  $(&quot;#menu&quot;).mouseover(function(){$(&quot;#menu&quot;).css(&quot;background-color&quot;,&quot;#01939a&quot;)});

  $(&quot;#menu&quot;).mouseout(function(){$(&quot;#menu&quot;).css(&quot;background-color&quot;,&quot;#006064&quot;)});
```

});

Попробовать в редакторе

Анимация в jQuery

С помощью метода **animate()** Вы можете создавать на Ваших страницах полноценную анимацию.

Синтаксис:

$("селектор").animate({стили},скорость,функция_смягчения,функция);

стили задает CSS стили для анимации (к элементу одновременно может быть применено несколько стилей).

скорость задает скорость анимации. Вы можете указать скорость используя предопределенные свойства: "slow", "fast", "normal" (медленно, быстро, нормально) или указать скорость в миллисекундах (1000 миллисекунд = 1 секунда).

функция_смягчения задает функцию, которая будет отвечать за плавность выполнения анимации.

функция указывает имя функции, код которой будет выполнен после завершения анимации.

Пример

```
$(document).ready(function(){

  $(&quot;#but1&quot;).click(function(){
    $(&quot;p&quot;).animate({fontSize:30},2000);
    $(&quot;p&quot;).animate({top:220},2000);
    $(&quot;p&quot;).animate({fontSize:&quot;1em&quot;},2000);
    $(&quot;p&quot;).animate({left:320},2000);
    $(&quot;p&quot;).animate({top:0,left:0},2000);
  });
```

```
});
```

Попробовать в редакторе

Обратите внимание: для того, чтобы воспользоваться свойствами позиционирования элемента необходимо вначале установить свойству position значение отличное от static.

Обратите внимание: полный список всех существующих эффектов jQuery с примерами их использования Вы найдете в нашем jQuery справочнике .

Работа с DOM

jQuery содержит набор методов значительно упрощающих взаимодействие с DOM.

Методы, которые будут рассмотрены в данной главе могут применяться и для HTML и для XML документов.

Изменение содержимого элементов с помощью jQuery

С помощью метода **html()** Вы можете изменить или узнать внутреннее содержимое выбранного элемента.

Синтаксис:

```
//Узнаем содержимое выбранного элемента
```

$("селектор").html();

//Изменим содержимое выбранного элемента

$("селектор").html("новое содержимое");

Пример

```
$(document).ready(function(){

  $(&amp;quot;#but1&amp;quot;).click(function(){

    $(&amp;quot;#par1&amp;quot;).html(&amp;quot;<b>jQuery</b>  - это JavaScript библиотека, значитель
но

    упрощающая написание кода.&amp;quot;);

  });
```

```
  $(&quot;#but2&quot;).click(function(){

    $(&quot;#par2&quot;).html(&quot;jQuery  значительно облегчает взаимодействие с DOM.&quot;);

  });

  $(&quot;#but3&quot;).click(function(){

    alert($(&quot;#par1&quot;).html());

  });

  $(&quot;#but4&quot;).click(function(){

    alert($(&quot;#par2&quot;).html());

  });



});
```

Попробовать в редакторе

Быстрый просмотр

С помощью метода **append()** Вы можете вставить произвольный текст после внутреннего содержимого выбранного элемента.

С помощью метода **prepend()** Вы можете вставить произвольный текст перед внутренним содержимым выбранного элемента.

Синтаксис:

//Добавим текст после внутреннего содержимого элемента

$("селектор").append("произвольный текст");

//Добавим текст перед внутренним содержимым элемента

$("селектор").prepend("произвольный текст");

Пример

$(document).ready(function(){

```
  $(&quot;#but1&quot;).click(function(){

    $(&quot;#par1&quot;).prepend(&quot;<b>jQuery</b>  - это  &quot;);

  });

  $(&quot;#but2&quot;).click(function(){

    $(&quot;#par1&quot;).append(&quot;  значительно упрощающая написание
кода.&quot;);

  });

  $(&quot;#but3&quot;).click(function(){

    $(&quot;body&quot;).append(&quot;<p>Я добавленный абзац.</p>&quot;);

  });

});
```

Попробовать в редакторе

Быстрый просмотр

С помощью метода **attr()** Вы можете узнать или изменить содержимое указанного атрибута у выбранного элемента.

С помощью метода **removeAttr()** Вы можете удалить указанный атрибут у выбранного элемента.

//Узнаем значение произвольного атрибута

$("селектор").attr("атрибут");

//Установим новое значение произвольному атрибуту

$("селектор").attr("атрибут", "новое значение");

//Удалим атрибут

$("селектор").removeAttr("атрибут");

Пример

```
$(document).ready(function(){




  $(&quot;#but1&quot;).click(function(){

    alert($(&quot;#anchor1&quot;).attr(&quot;href&quot;));

  });

  $(&quot;#but2&quot;).click(function(){

    $(&quot;#anchor1&quot;).attr(&quot;href&quot;,&quot;http://www.wisdomweb.ru/JQ/&quot;);


  });

  $(&quot;#but3&quot;).click(function(){

    $(&quot;#anchor1&quot;).removeAttr(&quot;href&quot;);

  });
```

});

Попробовать в редакторе

Быстрый просмотр

Метод **wrap** позволяет "обернуть" выбранный элемент указанными тэгами.

$("селектор").wrap("<нач_тэг><кон_тэг>");

Пример

$(document).ready(function(){

$("#but1").click(function(){

$("#par1").wrap("");

```
  });

  $(&quot;#but2&quot;).click(function(){

     $(&quot;#par2&quot;).wrap(&quot;<b></b>&quot;);

  });

  $(&quot;#but3&quot;).click(function(){

     $(&quot;#par3&quot;).wrap(&quot;<div  id='wrap1'></div>&quot;);

  });



});
```

Попробовать в редакторе

Обратите внимание: полный список всех существующих методов jQuery для работы с DOM с примерами использования Вы найдете в нашем jQuery справочнике
.

Методы манипулирования DOM

С помощью методов перечисленных в таблице ниже Вы можете производить различные манипуляции над DOM.

Метод	Описание
add()	Добавляет указанный элемент к группе выбранных элементов.
addClass()	Добавляет указанный класс или классы выбранному элементу.
after()	Вставляет указанное содержимое после выбранного элемента.
append()	

Вставляет указанное содержимое в конце содержимого выбранного элемента.

appendTo()

Вставляет указанное содержимое в конце содержимого выбранного элемента.

attr()

Вставляет новое или возвращает текущее значение указанного атрибута выбранного элемен

before()

Вставляет указанное содержимое перед выбранным элементом.

children()

Возвращает все элементы потомки указанного элемента.

contents()

Возвращает все элементы потомки указанного элемента включая текстовые узлы и узлы ком

clone()

Позволяет скопировать выбранный элемент.

detach()

Удаляет выбранный элемент (при этом сохраняя его копию).

empty()

Удаляет содержимое выбранного элемента и всех его элементов потомков.

eq()

Позволяет выбрать элемент из группы элементов по его порядковому номеру.

first()

Возвращает первый элемент из группы выбранных элементов.

hasClass()

Проверяет содержит ли выбранный элемент класс с указанным именем.

html()

Устанавливает новое или возвращает текущее содержимое выбранного элемента.

insertAfter()

Вставляет новый (или перемещает существующий) элемент после выбранного элемента.

insertBefore()

Вставляет новый (или перемещает существующий) элемент перед выбранным элементом.

last()

Возвращает последний элемент из группы выбранных элементов.

next()

Возвращает следующий элемент брат за выбранным элементом.

nextAll()

Возвращает все элементы братья, которые следуют за выбранным элементом.

parent()

Возвращает родительский элемент выбранного элемента.

prev()

Возвращает предыдущий элемент брат выбранного элемента.

prevAll()

Возвращает все элементы братья, которые следуют перед выбранным элементом.

prepend()

Вставляет указанное содержимое в начало содержимого выбранного элемента.

prependTo()

Вставляет указанное содержимое в начало содержимого выбранного элемента.

remove()

Удаляет выбранный элемент.

removeAttr()

Удаляет указанный атрибут у выбранного элемента.

removeClass()

Удаляет у выбранного элемента один класс или более.

replaceAll()

Заменяет текущий элемент новым содержимым.

replaceWith()

Заменяет текущий элемент новым содержимым.

siblings()

Возвращает все элементы братья указанного элемента.

text()

Устанавливает новое или возвращает текущее текстовое содержимое выбранного элемента.

toggleClass()

Поочередно присваивает указанные классы выбранному элементу.

unwrap()

Удаляет родительский элемент выбранного элемента.

val()

Устанавливает новое или возвращает текущее содержимое атрибута value выбранного элемен

wrap()

Оборачивает каждый выбранный элемент указанным тэгом (или группой тэгов).

wrapAll()

Оборачивает группу выбранных элементов указанным тэгом (или группой тэгов).

wrapInner()

Оборачивает содержимое каждого выбранного элемента указанным тэгом (или группой тэгов)

Методы CSS манипулирования

С помощью методов перечисленных в таблице ниже Вы можете производить различные манипуляции над оформлением элементов.

Метод	Описание
addClass()	Добавляет указанный класс или классы выбранному элементу.
css()	Позволяет узнать текущее или установить новое значение указанному CSS свойству (или сво
hasClass()	Проверяет содержит ли выбранный элемент класс с указанным именем.
height()	Позволяет узнать текущую или установить новую высоту выбранному элементу.

offset()

Позволяет узнать текущее или установить новое местоположение выбранного элемента отно

position()

Позволяет узнать текущее местоположение выбранного элемента относительно родительског

removeClass()

Удаляет у выбранного элемента один класс или более.

scrollLeft()

Позволяет узнать текущую или установить новую позицию полосы прокрутки выбранного эле

scrollTop()

Позволяет узнать текущую или установить новую позицию полосы прокрутки выбранного эле

toggleClass()

Поочередно присваивает указанные классы выбранному элементу.

width()

Позволяет узнать текущую или установить новую ширину выбранному элементу.

Методы создания AJAX запросов

AJAX запросы - это асинхронные запросы к серверу позволяющие обновлять только ту часть страницы, которая содержит новую информацию, без необходимости обновлять страницу целиком.

Использование AJAX запросов ускоряет загрузку страниц и снимает нагрузку с сервера.

Все существующие в jQuery методы для создания AJAX запросов перечислены в таблице ниже:

Метод

Описание

$.ajax()

Выполняет AJAX запрос.

ajaxComplete()

Определяет функцию, код которой будет выполнен когда AJAX запрос будет совершен.

ajaxError()

Определяет функцию, код которой будет выполнен если во время выполнения AJAX запроса

ajaxSend()

Определяет функцию, код которой будет выполнен перед отправлением AJAX запроса на се

$.ajaxSetup()

Позволяет установить данные для будущих AJAX запросов.

ajaxStart()

Определяет функцию, код которой будет выполнен перед тем, как первый AJAX запрос из гр

ajaxStop()

Определяет функцию, код которой будет выполнен, когда последний AJAX запрос из группы

ajaxSuccess()

Определяет функцию, код которой будет выполнен если AJAX запрос будет совершен успешно

$.get()

Позволяет загрузить данные с сервера используя HTTP запрос GET.

$.getJSON()

Позволяет загрузить JSON - данные с сервера используя HTTP запрос GET.

$.getScript()

Позволяет загрузить с сервера JavaScript код и исполнить его.

load()

Позволяет загрузить данные с сервера и вставить их в содержимое выбранного HTML элеме

$.param()

Позволяет создать сериализованное представление массива или объекта.

$.post()

Позволяет загрузить данные с сервера используя HTTP запрос POST.

serialize()

Позволяет закодировать группу элементов формы как строку для отправки с помощью AJAX

serializeArray()

Позволяет закодировать группу элементов формы как массив из их имен и значений.

Эффекты jQuery

Эффекты представленные в таблице ниже позволяют скрывать и отображать элементы различными способами.

Метод	Описание
fadeIn()	Постепенно меняет прозрачность выбранного элемента делая его видимым.
fadeOut()	Постепенно меняет прозрачность выбранного элемента делая его невидимым.

fadeTo()

Постепенно меняет прозрачность выбранного элемента до указанного значения.

hide()

Скрывает выбранный элемент.

show()

Отображает скрытый элемент.

slideDown()

Постепенно изменяет высоту выбранного элемента делая его видимым.

slideToggle()

Применяет к выбранному элементу метод slideDown() если он невидим и slideUp() если он ото

slideUp()

Постепенно изменяет высоту выбранного элемента делая его невидимым.

Анимация jQuery

С помощью методов перечисленных в таблице ниже Вы можете создать на Ваших страницах анимацию.

Метод

Описание

animate()

Выполняет заданную анимацию для выбранного элемента.

clearQueue()

Очищает очередь функций выбранного элемента.

delay()

Устанавливает задержку вызова следующей функции в очереди выбранного элемента.

dequeue()

Вызывает следующую функцию в очереди выбранного элемента.

stop()

Останавливает выполнение анимации для выбранного элемента.

queue()

Позволяет добавлять функции в очередь выбранного элемента.

Остальные методы jQuery

В данном разделе перечислены различные методы jQuery, которые не попали ни в один предыдущий раздел.

Метод

Описание

data()

Позволяет привязать или извлечь привязанные ранее данные у выбранного элемента.

$.each()

Позволяет исполнить переданную функцию для каждого элемента переданного объекта или

each()

Позволяет исполнить переданную функцию для каждого выбранного элемента.

$.extend()

Позволяет объединить два или более переданных объектов в один объект.

filter()

Позволяет отфильтровать выбранные элементы.

get()

Позволяет извлечь элемент как DOM объект из группы выбранных элементов.

index()

Позволяет узнать позицию элемента среди элементов братьев (элементов находящихся на о

$.isArray()

Проверяет является ли переданный объект массивом.

$.isEmptyObject()

Проверяет является ли переданный объект пустым объектом.

$.isFunction()

Проверяет является ли переданный объект функцией.

$.isPlainObject()

Проверяет является ли переданный объектом простым объектом.

$.map()

Обрабатывает заданной функцией все значения переданного массива или объекта и создает

removeData()

Позволяет удалить привязанные к выбранному элементу данные.

size()

Является устаревшим в jQuery 1.8 Возвращает количество выбранных элементов.

Методы взаимодействия jQuery UI

С помощью методов взаимодействия Вы можете сделать элементы перетаскиваемыми, сортируемыми, выделяемыми и изменяющими размер.

Метод

Описание

draggable

Позволяет сделать выбранный элемент перетаскиваемым.

droppable

Определяет область для приема перетаскиваемого элемента.

resizable

Позволяет сделать выбранный элемент растягиваемым.

selectable

Позволяет сделать выбранные элементы выделяемыми.

sortable

Позволяет сделать выбранные элементы сортируемыми.

Виджеты jQuery UI

JQuery UI предоставляет набор готовых виджетов предназначенных для создания пользовательского интерфейса.

Внешний вид каждого виджета может быть настроен либо с помощью выбора одной из стандартных тем доступных при скачивании библиотеки, либо с помощью настройщика тем, либо вручную путем редактирования файла *jquery-ui-1.8.12.custom.css*.

Поведение каждого виджета может настраиваться с помощью передаваемых ему свойств и методов.

Виджет

Описание

accordion

Превращает выбранный элемент в виджет accordion.

autocomplete

Превращает выбранный элемент в виджет autocomplete.

button

Превращает выбранный элемент в стилизованную кнопку, внешний вид которой можно настр

datepicker

Превращает выбранный элемент в виджет datepicker.

dialog

Превращает выбранный элемент в диалоговое окно.

progressbar

Превращает выбранный элемент в полосу загрузки.

slider

Превращает выбранный элемент в виджет slider.

tabs

Превращает выбранный элемент в виджет tabs.

Обработчики событий jQuery

С помощью методов перечисленных в таблице ниже Вы можете создавать обработчики событий и привязывать их к элементам.

Синтаксис:

//Привяжем к кнопке обработчик события click

$(":button").click(function(){$("#par1").html("Новый текст")})

//Вызовем привязанный ранее обработчик

$(":button").click()

Обратите внимание: для того, чтобы узнать о желаемом методе или событие больше щелкните на его названии мышкой.

Метод

Описание

blur()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент пе

change()

Привязывает или вызывает функцию, код которой будет выполнен в случае изменения содер

click()

Привязывает или вызывает функцию, код которой выполнится, когда на выбранном элементе

dblclick()

Привязывает или вызывает функцию, код которой выполнится, когда на выбранном элементе

error()

Является устаревшим в jQuery 1.8. Привязывает или вызывает функцию, код которой выполнится если при

focus()

Привязывает или вызывает функцию, код которой будет выполнен, когда выбранный элемент

focusin()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент или

focusout()

Привязывает или вызывает функцию, код которой выполнится, когда выбранный элемент или

hover()

Привязывает одну или две функции к выбранному элементу. Код первой привязанной функции

keydown()

Привязывает или вызывает функцию, код которой выполнится, когда на клавиатуре будет на

keyup()

Привязывает или вызывает функцию, код которой выполнится, когда нажатая на клавиатуре

load()

Является устаревшим в jQuery 1.8+ Привязывает или вызывает функцию, код которой будет выполнен посл

mousedown()

Привязывает или вызывает функцию, код которой будет выполнен после нажатия клавиши мы

mouseenter()

Привязывает или вызывает функцию, код которой будет выполнен, когда на выбранный элеме

mouseleave()

Привязывает или вызывает функцию, код которой будет выполнен, когда указатель мыши буд

mousemove()

Привязывает или вызывает функцию, код которой будет выполнен при передвижении указате

mouseout()

Привязывает или вызывает функцию, код которой будет выполнен, когда указатель мыши буд

mouseover()

Привязывает или вызывает функцию, код которой будет выполнен, когда на выбранный элеме

mouseup()

Привязывает или вызывает функцию, код которой будет выполнен, когда нажатая кнопка мыш

ready()

Привязывает или вызывает функцию, код которой будет выполнен, когда страница будет полн

resize()

Привязывает или вызывает функцию, код которой будет выполнен при изменении размера окн

scroll()

Привязывает или вызывает функцию, код которой будет выполнен при прокрутке содержимог

select()

Привязывает или вызывает функцию, код которой будет выполнен при выделении текста вы

submit()

Привязывает или вызывает функцию, код которой будет выполнен при отправлении содержим

unload()

Является устаревшим в jQuery 1.8+ Привязывает или вызывает функцию, код которой будет выполнен при в

Управление обработчиками событий

С помощью методов представленных в таблице ниже Вы сможете управлять обработчиками событий.

Метод

Описание

bind()

Привязывает к выбранному элементу один обработчик события или более.

delegate()

Добавляет один обработчик события или более к элементам потомкам выбранного элемента.

one()

Привязывает к выбранному элементу один обработчик события или более. Привязанные обра

toggle()

Позволяет привязать к выбранному элементу несколько обработчиков событий, между вызов

trigger()

Вызывает указанный обработчик события у выбранного элемента.

triggerHandler()

Вызывает указанный обработчик события у выбранного элемента.

unbind()

Удаляет у выбранных элементов обработчики событий, которые были привязаны с помощью

undelegate()

Удаляет у выбранных элементов обработчики событий, которые были привязаны с помощью

Объект event

Объект event хранит информации о произошедшем событии.

Объект event создается для каждого произошедшего события, но для того чтобы иметь возможность обратится к его свойствам и методам, его необходимо явно передать в обработчик события.

Синтаксис:

```
$(селектор).событие(function(event){

  //Затем в коде обработчика Вы можете обращаться к его свойствам и методам следующим образом:

  event.data;
```

```
  event.preventDefault();

});
```

Обратите внимание: при передаче объекта event в обработчик Вы можете использовать любое имя.

Свойство	Описание
currentTarget	Содержит имя DOM элемента, в котором произошло событие.
data	Содержит дополнительные данные переданные обработчику события во время привязки его к
pageX	

Содержит координаты указателя мыши по оси X во время вызова события.

pageY

Содержит координаты указателя мыши по оси Y во время вызова события.

result

Содержит последнее значение возвращенное вызванным ранее обработчиком события.

target

Содержит имя DOM элемента, который вызвал событие.

timeStamp

Содержит количество прошедших с 1 Января 1970 года миллисекунд до вызова данного собы

type

Содержит тип (название) произошедшего события.

which

Содержит код кнопки, которая была зажата во время вызова данного события.

Метод

Описание

isDefaultPrevented()

Позволяет узнать был ли вызван метод preventDefault() для данного элемента.

isImmediatePropagationStopped()

Позволяет узнать был ли вызван метод stopImmediatePropagation() для данного элемента.

isPropagationStopped()

Позволяет узнать был ли вызван метод stopPropagation() для данного элемента.

preventDefault()

Предотвращает выполнение стандартного действия элемента.

stopImmediatePropagation()

Запрещает вызов остальных обработчиков события привязанных к элементу.

stopPropagation()

Останавливает "всплытие" вызова события к родительским элементам.

Плагины jQuery

Реализация одних и тех же функций в различных приложениях побуждает разработчиков заново писать один и тот же код несколько раз лишь незначительно изменяя его под конкретное приложение.

Плагины jQuery позволяют забыть разработчикам о данной проблеме. Разработчик может один раз написать плагин, который позволяет реализовать определенную функцию и затем использовать его в необходимых приложениях написав только одну строчку кода.

В интернете можно найти огромное количество бесплатных jQuery плагинов написанных другими разработчиками и использовать их для создания своих приложений. Можете начать поиски интересных jQuery плагинов на сайте PluginDetector.com .

Создание плагина

Для того, чтобы создать плагин необходимо добавить к объекту jQuery.fn свойство, имя которого будет является именем плагина:

Синтаксис:

```
//Определяем код плагина

(function($){

  $.fn.имяПлагина  = function() {

    // Код  плагина

  };
```

})(jQuery);

//Вызываем плагин

$(селектор).имяПлагина();

Обратите внимание: обертка *(function($){ }) (jQuery)* используется здесь для того, чтобы внутри кода плагина можно было использовать знак $ не боясь при этом, что он будет конфликтовать с другими библиотеками JavaScript.

Ключевое слово this в плагинах

При написании плагинов следует обратить внимание на то, что в коде плагина ключевое слово this будет относиться к jQuery объекту, который вызвал этот плагин, а не к DOM объекту как в функциях обратного вызова.

Это значит, что в коде плагина для того, чтобы обратится к jQuery объекту необходимо использовать this, а в коде функций обратного вызова использовать $(this).

Пример

(function($){

/* Создаем плагин с именем plugin, который будет находить сумму содержимого всех

```
выбранных  абзацев на странице */

 $.fn.plugin=function(){

    var sum=0;

    /* Для того,  чтобы сослаться на выбранный элемент в коде плагина мы

      используем this, но в функции обратного вызова мы уже должны  использовать

       $(this)  для этих же целей */

    this.each(function(){

       sum=sum+Number($(this).html());

    });

    //  Возвращаем полученную сумму

    return sum;
```

```
  }
```

```
})(jQuery)
```

Попробовать в редакторе

Поддержка цепочки методов

Плагин в предыдущем примере возвращал целое значение, но часто плагины используются для модификации группы элементов и передачи их следующему методу в цепочке.

Пример цепочки методов:

```
//Пример цепочки из двух методов css и html
```

```
$(&quot;p&quot;).css(&quot;color&quot;,&quot;red&quot;).html(&quot;Новое содержимое&quot;);
```

Для того, чтобы Ваш плагин не "разрывал" цепочку методов необходимо, чтобы он возвращал ключевое слово this.

Пример

```
(function($){

  /* Создаем  плагин с именем adjust, который будет устанавливать размер текста

  выбранных элементов  равным 1.4em и передавать их дальше по цепочки методов. */

  $.fn.adjust=function(){

  /* Для того,  чтобы поддержать цепочку методов нужно вернуть из плагина

 преобразованную группу выбранных элементов */

    return this.each(function(){

      $(this).css(&quot;fontSize&quot;,&quot;1.4em&quot;);

    });

  }
```

```
})(jQuery)
```

Попробовать в редакторе

Опции плагинов

Многие плагины позволяют пользователям настраивать их поведение с помощью передаваемых им опций.

Пример

```
(function($){

/* Создаем плагин с именем adjust, который будет устанавливать указанный размер и
 цвет выбранным элементам. */
  $.fn.adjust=function(option){
   /* Установим значения опций по умолчанию. Они будут использованы если
```

```
пользователь

  при вызове  плагина не передаст значений. */

    var setting={size:'1.4em',color:'red'};

    return this.each(function(){

      // Если  пользователь передал опции соединить их с опциями по умолчанию

      if (option){$.extend(setting,option);}

      $(this).css(&amp;quot;fontSize&amp;quot;,setting.size);

      $(this).css(&amp;quot;color&amp;quot;,setting.color);

    });

  }



})(jQuery)
```

Попробовать в редакторе

Вынесение кода плагина во внешний файл

Заключительным шагом в создании плагинов должно стать вынесение кода плагина в отдельный файл. И написания инструкций, которые должны разъяснить потенциальным пользователям как и зачем можно использовать Ваш плагин.

Пример

<script src="http://www.wisdomweb.ru/editor/plugin1.js"></script>

$(document).ready(function(){

  $("#but1").click(function(){

    $("#demo p").adjust();

  });

```
  $(&quot;#but2&quot;).click(function(){

    $(&quot;#demo p&quot;).adjust({size:&quot;2em&quot;});

  });

  $(&quot;#but3&quot;).click(function(){

    $(&quot;#demo p&quot;).adjust({size:&quot;1.7em&quot;,color:&quot;green&quot;});

  });



});
```

jQuery UI

jQuery UI представляет собой группу плагинов jQuery облегчающих создание интерфейса веб-приложений.

Пример

$(document).ready(function(){ $("#drag").draggable(); $("#sortable").sortable();

```
$
(
&quot;#sortable&quot;
).
disableSelection
();
```

```
$
(
&quot;#datepicker&quot;
).
datepicker
({
monthNames
:
```

```
[
&quot;Январь&quot;
,
&quot;Февраль&quot;
,
&quot;Март&quot;
,
&quot;Апрель&quot;
,
&quot;Май&quot;
,
&quot;Июнь&quot;
,
&quot;Июль&quot;
,
&quot;Август&quot;
,
```

```
&quot;Сентябрь&quot;
,
&quot;Октябрь&quot;
,
```

"Ноябрь"
,
"Декабрь"
],

dayNamesMin
:
[
"Пн"
,
"Вт"
,
"Ср"
,
"Чт"
,
"Пт"
,
"Сб"
,
"Вс"
]});

});

Попробовать в редакторе

Подключение jQuery UI

Для того, чтобы воспользоваться возможностями плагинов jQuery UI их необходимо вначале подключить к странице, на которой они будут использоваться.

Существуют два варианта подключения jQuery UI:

1. Локальное подключение . Данный способ требует скачивание специального файла с официального сайта;
2. Удаленное подключение . Данный способ не требует скачивание файла, а вместо этого использует его удаленно (*данная услуга*

предоставляется компанией Google
).

Локальное подключение jQuery UI

На официальном сайте Вы можете или скачать стандартную комплектацию jQuery UI или собрать свою собственную.

Стандартная комплектация jQuery UI включает в себя все существующие плагины и имеет стандартную тему оформления. Для того, чтобы скачать стандартную комплектацию просто перейдите на сайт jQuery UI и нажмите кнопку **Download**.

Если Вы хотите собрать собственную комплектацию jQuery UI Вам необходимо перейти на сайт jQuery UI и выполнить шаги перечисленные ниже (*пропустите эти шаги если Вы скачали стандартную комплектацию*
):

Шаг 1: Выбрать необходимые компоненты

По умолчанию в файл для скачивания включены все существующие плагины. Если какие-то из них не нужны Вы можете убрать галочку напротив их названия и сократить тем самым размер итогового файла (*стандартная комплектация jQuery UI имеет размер ~1мб*).

Обратите внимание: размер файла библиотеки влияет на скорость загрузки страницы, поэтому меньший размер всегда предпочтителен.

Шаг 2: Выбрать оформление

Выберите одну из стандартных тем оформления плагинов jQuery UI в поле Theme или создайте свою тему с помощью themeroller'a.

Шаг 3: Выбрать версию

Выберите версию jQuery UI в поле Version.

Обратите внимание: рекомендуем всегда выбирать последнюю доступную версию т.к. как более новые версии всегда содержат интересные нововведения.

Шаг 4: Скачать jQuery UI

Нажмите кнопку Download и сохраните файл в удобное для Вас место на жестком диске.

Теперь (*вне зависимости скачали ли Вы стандартную комплектацию или собрали свою собственную*) необходимо подключить jQuery UI к скрипту. Для этого необходимо распаковать скаченный файл и указать пути к файлам
jquery-ui-версия.custom.css
и
jquery-ui-версия.custom.min.js
в секции head скрипта.

Синтаксис:

<!-- 1. Подключим библиотеку jQuery (без нее jQuery UI не будет работать) --> <script src
=
"http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"
></script>

<!-- 2. Подключим jQuery UI -->

<link
type
=
"text/css"
href
=
"

путь
_
к
_
папке
/css/themename/jquery-ui-1.8.13.custom.css"
rel
=
"stylesheet"
/>

<script
type
=
"text/javascript"
src
=
"
путь
_
к
_
папке
/js/jquery-ui-1.8.13.custom.min.js"
></script>

Удаленное подключение jQuery UI

В этом варианте подключения Вы не можете настраивать комплектацию jQuery UI и можете использовать только ее стандартную версию.

Для того, чтобы подключить библиотеку удаленно необходимо добавить следующие строчки в секцию head Вашей страницы:

Синтаксис:

<!-- 1. Подключим библиотеку jQuery (без нее jQuery UI не будет работать) --> <script sr
c =
"http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"
></script>

```
<!-- 2.
Подключим
jQuery UI -->

<link
href
=
&quot;http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/themes/base/jquery-ui.css&quot;
rel
=
&quot;stylesheet&quot;
type
=
&quot;text/css&quot;
/>

<script
src
=
&quot;http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/jquery-ui.min.js&quot;
></script>
```

Методы взаимодействия

С помощью методов взаимодействия Вы можете создавать элементы, которые можно перетаскивать мышкой, которые могут изменять размер, которые можно сортировать и которые можно выделять мышкой.

Перетаскиваемые элементы

С помощью метода **draggable** Вы можете сделать выбранный элемент перетаскиваемым.

Синтаксис

$("селектор").draggable({опция1:значение1, опцияN:значениеN})

С помощью опций, которые передаются методу draggable Вы можете настраивать дополнительные аспекты поведения перетаскиваемых элементов.

Всем перетаскиваемым элементам назначается класс **ui-draggable** и дополнительно класс **ui-draggable-dragging** во время перетаскивания.

Пример

```
$(document).ready(function()  {

   $(&amp;quot;div&amp;quot;).draggable();

});
```

Попробовать в редакторе

С помощью опции **axis** Вы можете определить направление оси, по которой можно перетаскивать элемент.

Пример

```
$(document).ready(function()  {

   $(&quot;#drag1&quot;).draggable({axis:&quot;x&quot;});

   $(&quot;#drag2&quot;).draggable({axis:&quot;y&quot;});

});
```

Попробовать в редакторе

С помощью опции **cursor** Вы можете задать вид, который будет принимать указатель мыши во время перетаскивания элемента.

Пример

```
$(document).ready(function()  {

   $(&quot;#drag1&quot;).draggable({cursor:&quot;move&quot;});

   $(&quot;#drag2&quot;).draggable({cursor:&quot;help&quot;});

});
```

Попробовать в редакторе

С помощью опции **helper** Вы можете определить вид элемента-помощника. Элемент-помощник отображается во время перетаскивания элемента. По умолчанию во время перетаскивания отображается сам перетаскиваемый элемент.

Пример

```
$(document).ready(function() {

  $(&quot;#drag1&quot;).draggable({helper:&quot;clone&quot;});

  $(&quot;#drag2&quot;).draggable({helper:function(event){return $(&quot;<div>Я элемент

  помощник.</div>&quot;)}});

});
```

Попробовать в редакторе

Опция **revert** определяет будет ли элемент возвращаться на изначальную позицию после завершения перетаскивания.

Пример

```
$(document).ready(function()  {

   $(&amp;quot;#drag1&amp;quot;).draggable({revert:true});

});
```

Попробовать в редакторе

Области для перетаскиваемых элементов

С помощью метода **droppable** Вы можете определить область, в которую можно переместить перетаскиваемый элемент.

Синтаксис

```
$(&amp;quot;селектор&amp;quot;).droppable({опция1:значение1, опцияN:значениеN})
```

С помощью опций, которые передаются методу droppable Вы можете настраивать дополнительные аспекты поведения области для перетаскиваемых элементов.

Код функции переданный, в опции **drop** выполнится, когда перетаскиваемый элемент будет перетащен в область и отпущен.

Код функции переданный, в опции **over** выполнится, когда перетаскиваемый элемент будет заведен в границы области.

Пример

```
$(document).ready(function(){

  $(&quot;#drag1&quot;).draggable({stack:&quot;#drop1&quot;});

  $(&quot;#drop1&quot;).droppable({

    over:function(){

      $(&quot;#drag1&quot;).css(&quot;background-color&quot;,&quot;#d7fa99&quot;);

      $(&quot;#drop1&quot;).css(&quot;background-color&quot;,&quot;#d7fa99&quot;);

    },

    drop:function(){

      $(&quot;#drag1&quot;).css(&quot;display&quot;,&quot;none&quot;);

      $(&quot;#drop1&quot;).css(&quot;background-color&quot;,&quot;#c4f66f&quot;);
```

```
        $(&quot;#drop1&quot;).html(&quot;Перемещение завершено успешно.&quot;);

        alert(&quot;Вы  перетащили элемент с id=drag1 в область с id=drop1.&quot;);

      }

   });

});
```

Попробовать в редакторе

Быстрый просмотр

С помощью опции **accept** Вы можете указать элементы, которые будут приниматься областью.

Пример

```
$(document).ready(function(){

   $(&quot;#drag1&quot;).draggable({stack:&quot;#drop1&quot;});

   $(&quot;#drag2&quot;).draggable({stack:&quot;#drop1&quot;,revert:true});
```

```
$(&quot;#drop1&quot;).droppable({

  accept:&quot;#drag1&quot;,

  over:function(){

    $(&quot;#drag1&quot;).css(&quot;background-color&quot;,&quot;#d7fa99&quot;);

    $(&quot;#drop1&quot;).css(&quot;background-color&quot;,&quot;#d7fa99&quot;);

  },

  drop:function(){

    $(&quot;#drag1&quot;).css(&quot;display&quot;,&quot;none&quot;);

    $(&quot;#drop1&quot;).css(&quot;background-color&quot;,&quot;#c4f66f&quot;);

    $(&quot;#drop1&quot;).html(&quot;Перемещение завершено успешно.&quot;);

  }
```

```
  });

});
```

Попробовать в редакторе

Быстрый просмотр

Растягиваемые элементы

С помощью метода **resizable** Вы можете сделать элементы растягиваемыми.

Синтаксис

```
$(&quot;селектор&quot;).resizable({опция1:значение1, опцияN:значениеN})
```

С помощью опций, которые передаются методу resizable Вы можете настраивать дополнительные аспекты поведения растягиваемых элементов.

Пример

```
$(document).ready(function(){

  $(&quot;#resize1&quot;).resizable();
```

});

Попробовать в редакторе

Быстрый просмотр

Если опция **animate** имеет значение true размер элемента будет изменятся анимировано.

С помощью опции **helper** Вы можете задать вид помощника, с помощью которого пользователь будет узнавать о текущем размере окна во время растягивания.

Пример

```
$(document).ready(function(){

  $(&quot;#resize1&quot;).resizable({animate:true,helper:&quot;ui-state-highlight&quot;});

});
```

Попробовать в редакторе

Быстрый просмотр

Выделяемые элементы

С помощью метода **selectable** Вы можете превратить группу элементов в список выбора.

Синтаксис

$("селектор").selectable({опция1:значение1, опцияN:значениеN})

С помощью опций, которые передаются методу selectable Вы можете настраивать дополнительные аспекты поведения выделяемых элементов.

С помощью классов **.ui-selecting** и **.ui-selected** Вы можете настроить оформление выбираемых и выбранных элементы.

Пример

```
$(document).ready(function(){

  $(&amp;quot;#select1&amp;quot;).selectable();

});
```

Попробовать в редакторе

Быстрый просмотр

Сортируемые элементы

С помощью метода **sortable** Вы можете сделать элементы списка сортируемыми.

Синтаксис

$("селектор").sortable({опция1:значение1, опцияN:значениеN})

С помощью опций, которые передаются методу sortable Вы можете настраивать дополнительные аспекты поведения сортируемых списков.

Пример

```
$(document).ready(function(){

  $(&amp;quot;#sort1&amp;quot;).sortable();

});
```

Попробовать в редакторе

Быстрый просмотр

Вы можете связать два сортируемых списка. Элементы одного связанного списка можно перемещать в другой связанный список.

Для того, чтобы воспользоваться этой возможностью необходимо задать связываемым спискам одинаковый класс и указать его в опции **connectWith**.

Пример

```
$(document).ready(function(){

  $(&quot;#sort1,#sort2&quot;).sortable({connectWith:&quot;.connect&quot;});

});
```

Попробовать в редакторе

Быстрый просмотр

Обратите внимание: узнать больше о методах взаимодействия Вы можете в нашем jQuery UI справочнике
.

Виджеты jQuery UI

jQuery UI предоставляет набор готовых виджетов предназначенных для создания пользовательского интерфейса веб-приложений.

Поведение виджетов может настраиваться с помощью передаваемых им опций.

Внешний вид виджетов может быть настроен:

1. С помощью выбора одной из стандартных тем доступных при скачивании библиотеки;
2. С помощью themeroller'a ;
3. Вручную путем редактирования файла jquery-ui-1.8.12.custom.css.

Общий вид создания виджетов jQuery UI выглядит примерно следующим образом:

1. Группировка элементов на странице особым образом (*индивидуально для каждого виджета*);
2. Применение к сгруппированным элементам специального метода, который превращает их в виджет.

Виджет accordion

Виджет accordion представляет собой группу объединенных выпадающих меню из которых только одно может быть отрыто одновременно.

Данный виджет используется для группировки информации на странице с целью экономии места.

Синтаксис:

/* 1. Группируем элементы: */

** **

// 1.1. Определяем заголовок элемента

<h2>Заголовок 1</h2>

/* 1.2. Определяем содержимое элемента (значение атрибута id содержимого должно

совпадать со значением href заголовка) */

** Содержимое 1 **

** **

// 2. Применяем метод к сгруппированным элементам:

$("#accordion").accordion({опция1:значение1,опцияN:значениеN});

Обратите внимание: с помощью опций, передающихся методу, Вы можете настраивать дополнительные аспекты поведения виджетов. Виджетам может быть передано сразу несколько опций.

Пример

```
$(document).ready(function() {

   $(&amp;quot;#accordion&amp;quot;).accordion();

});
```

Попробовать в редакторе

Виджет autocomplete

Виджет autocomplete позволяет ускорить ввод данных в поле, отображая по мере введения определенные предположения. Выбрав одно из предположений пользователь может автоматически завершить ввод.

Предположения выводится в случае, если данные введенные пользователем совпадают со значением одного из элементов из списка данных.

Вы можете подключать к виджету как локальные (*т.е. определенные в скрипте на этой же странице*) так и удаленные списки (*т.е. находящиеся на удаленном сервере*).

Локальные списки удобны для хранения небольших наборов данных (*например список*

улиц города), а
удаленные списки подходят для хранения больших наборов данных (
например база данных всех городов мира
).

Синтаксис:

/* 1. Определяем элемент input, в который будет производиться ввод информации */

<input id='auto' />

/* 2. Превращаем input в виджет autocomplete */

$('#auto').autocomplete({опция1:значение1,опцияN:значениеN})

Подключить к виджету список данных можно с помощью опции **source**.

С помощью опции **minLength** Вы можете указать минимальное количество символов, которое должно содержать поле ввода прежде, чем поиск совпадений начнет выполнятся.

Пример

$(document).ready(function() {

$("#auto1").autocomplete({source:["Дмитрий","Мария",&q

uot;Владимир","Алексей","Екатерина","Олег","Ольга"]});

});

Попробовать в редакторе

Виджет datepicker

Виджет **datepicker** привязывает к текстовому полю интерактивный календарь, который отображается когда поле становится активным.

Если пользователь щелкнет на какую-либо дату в календаре она будет автоматически введена в текстовое поле как будто он ввел ее вручную.

Синтаксис:

// 1. Создадим текстовое поле

<input type='text' id='datepicker' />

// 2. Привяжем к нему виджет datepicker

$('#datepicker').datepicker({опция1:значение1, опцияN:значениеN});

Пример

```
$(document).ready(function()  {

   $(&quot;#datepicker&quot;).datepicker();

   $(&quot;#datepicker1&quot;).datepicker({

      monthNames:[&quot;Январь&quot;,&quot;Февраль&quot;,&quot;Март&quot;,&quot;Апре
ль                                                                    &q
uot;,&quot;
Май
&quot;,&quot;
Июнь
&quot;,&quot;
Июль
&quot;,&quot;
Август
&quot;,&quot;
Сентябрь
&quot;,&quot;
Октябрь
&quot;,&quot;
Ноябрь
&quot;,&quot;
Декабрь
&quot;],

      dayNamesMin:[&quot;Вс&quot;,&quot;Пн&quot;,&quot;Вт&quot;,&quot;Ср&quot;,&quot;Чт
&quot;,&quot;
Пт
&quot;,&quot;
```

Сб
"],

firstDay:1,

dateFormat:"dd.mm.yy"

});

});

Попробовать в редакторе

Оформление кнопок

С помощью метода **button** Вы можете стилизовать:

- обычные кнопки (определяемые тэгами button и input type='button')
- кнопки отправления форм
- радио кнопки
- флажки
- ссылки

Синтаксис:

/* Если Вы хотите оформить только один элемент, необходимо выбрать его с помощью

селектора и вызвать метод button: */

$("селектор").button({опция1:значение1, опцияN:значениеN});

/* Если Вы оформляете группу элементов, необходимо вначале сгруппировать элементы

следующим образом: */

** **

<input type="radio" name="radio" id="value1" />

<label for="value1">Радио кнопка 1</label>

<input type="radio" name="radio" id="value2" />

<label for="value2">Радио кнопка 2</label>

<input type="radio" name="radio" id="value3" />

<label for="value3">Радио кнопка 3</label>

** **

// И затем вызвать метод buttonset:

$("#groupradio").buttonset({опция1:значение1, опцияN:значениеN});

Пример

```
$(document).ready(function()  {

   $(&amp;quot;#but1,#but2,#but3,#but4&amp;quot;).button();

   $(&amp;quot;#group1,#group2&amp;quot;).buttonset();

});
```

Попробовать в редакторе

В следующей главе будут рассмотрены оставшиеся виджеты имеющиеся в jQuery UI.

Обратите внимание: узнать больше о виджетах разобранных в данной главе Вы можете в нашем jQuery UI справочнике .

Эффекты jQuery UI

Возможность применения различных эффектов к элементам была и в jQuery. С помощью эффектов jQuery Вы, например, могли заставить абзац постепенно пропадать или появляться, применять к элементам анимацию и многое другое.

Эффекты jQuery UI значительно расширяют эти возможности. Теперь можно заставить элементы пропадать или появляться пятнадцатью разными способами.

В анимации Вы можете использовать свойства изменения цвета фона, текста и границ элементов.

jQuery UI также расширяет базовую функциональность jQuery для манипулирования классами оформления.

Скрытие и отображение элементов с помощью эффектов jQuery UI

В jQuery UI можно использовать 15 различных эффектов для скрытия и отображения элементов на страницах.

Эффекты в jQuery UI могут использоваться со следующими методами:

- **effect** - позволяет применяет указанный эффект к элементу
- **hide** - позволяет скрыть элемент с указанным эффектом
- **show** - позволяет отобразить элемент с указанным эффектом

Синтаксис:

effect(эффект,опции,скорость,функция)

hide(эффект,опции,скорость,функция)

show(эффект,опции,скорость,функция)

эффект указывает название эффекта, который будет применен. Некоторые эффекты (такие как scale, transfer и size) для своего вызова требуют задания дополнительных *опций*
.

скорость указывает скорость применения эффекта в миллисекундах (1000 миллисекунд = 1 секунда).

функция задает функцию, код которой будет выполнен после завершения применения эффекта.

Эффекты jQuery UI приведены в таблице ниже:

Название эффекта

Описание

Blind

Эффект "Ослепление".

Bounce

Эффект "Отскок".

Clip

Эффект "Отсекание".

Drop

Эффект "Падение".

Explode

Эффект "Взрыв".

Fade

Эффект "Выцветание".

Fold

Эффект "Складка".

Highlight

Эффект "Освещение".

Puff

Эффект "Рассеивание".

Pulsate

Эффект "Пульсирование".

Scale

Эффект "Масштабирование". С помощью него можно задать на сколько процентов от текущ

Shake

Эффект "Тряска".

Slide

Эффект "Скольжение".

Size

Эффект "Калибровка". С помощью исключенных в нее опций **width:ширина_в_пикселях**

Transfer

Эффект "Перехода". С помощью опций **to:id_или_Class_элемента** вы можете указать **className:имя_класса** Вы м

Увидеть все эффекты не требующие дополнительных опций для вызова в действии можно в следующем примере:

Пример

```
$(document).ready(function()  {

   $(&quot;#but1&quot;).click(function(){

      $(&quot;#testcontainer&quot;).hide($(&quot;#effect1&quot;).val(),{},1000  );

   });

   $(&quot;#but2&quot;).click(function(){

      $(&quot;#testcontainer&quot;).show($(&quot;#effect1&quot;).val(),{},1000  );

   });

});
```

Попробовать в редакторе

В следующем примере Вы можете увидеть использование эффектов требующих для вызова дополнительных опций.

Пример

```
$(document).ready(function()  {
```

```
$(&quot;#effect1&quot;).change(function(){

  if ($(&quot;#effect1&quot;).val()===&quot;scale&quot;){

    $(&quot;#scalecont&quot;).css(&quot;display&quot;,&quot;block&quot;);

    $(&quot;#transfercont&quot;).css(&quot;display&quot;,&quot;none&quot;);

    $(&quot;#sizecont&quot;).css(&quot;display&quot;,&quot;none&quot;);}

  else if ($(&quot;#effect1&quot;).val()===&quot;transfer&quot;){

    $(&quot;#transfercont&quot;).css(&quot;display&quot;,&quot;block&quot;);

    $(&quot;#scalecont&quot;).css(&quot;display&quot;,&quot;none&quot;);

    $(&quot;#sizecont&quot;).css(&quot;display&quot;,&quot;none&quot;);}

  else if ($(&quot;#effect1&quot;).val()===&quot;size&quot;){

    $(&quot;#sizecont&quot;).css(&quot;display&quot;,&quot;block&quot;);
```

```
    $(&quot;#transfercont&quot;).css(&quot;display&quot;,&quot;none&quot;);

    $(&quot;#scalecont&quot;).css(&quot;display&quot;,&quot;none&quot;);}

});

$(&quot;#but1&quot;).click(function(){

  var options={};

  if ($(&quot;#effect1&quot;).val()===&quot;scale&quot;){

    var options={percent:50};

    options.percent=$(&quot;#scalepercent&quot;).val();}

  else if ($(&quot;#effect1&quot;).val()===&quot;transfer&quot;){

    var options={to:&quot;#but1&quot;,  className:&quot;transfer-effect&quot;}

    options.to=$(&quot;#trans1&quot;).val();

    options.className=$(&quot;#trans2&quot;).val();}
```

```
    else if ($(&quot;#effect1&quot;).val()===&quot;size&quot;){

      var options={to:{width:200,height:200}};

      options.to.width=$(&quot;#size1&quot;).val();

      options.to.height=$(&quot;#size2&quot;).val();}

    $(&quot;#testcontainer&quot;).effect($(&quot;#effect1&quot;).val(),  options ,1000);

  });

});
```

Попробовать в редакторе

jQuery UI анимация

jQuery UI расширяет возможности jQuery анимации. Теперь Вы можете создавать анимацию, которая может манипулировать цветом.

В анимации теперь могут быть использованы следующие CSS свойств:

- backgroundColor
- borderColor
- borderBottomColor
- borderLeftColor
- borderRightColor
- borderTopColor
- color
- outlineColor

Пример

```
$(document).ready(function() {

  $(&quot;#but1&quot;).click(function(){

    $(&quot;#testcontainer&quot;).animate({

      borderColor:&quot;#EA3B53&quot;,

      borderWidth:3,

       backgroundColor:&quot;#97D400&quot;,

       width:500,

       height:160},1500);
```

```
  });

});
```

Попробовать в редакторе

Манипуляции с CSS оформлением

Эффекты jQuery UI расширяют базовую функциональность jQuery для манипулирования с классами.

Теперь переключение между различными оформлениями элемента будет происходить анимировано.

Синтаксис:

```
/* Добавит новый класс элементу с анимированным  переходом между

различными состояниями оформления */
```

addClass(имя_класса, продолжительность_перехода)

```
/* Удалит указанный класс у элемента с анимированным  переходом между
```

различными состояниями оформления */

removeClass(имя_класса, продолжительность_перехода)

/* Удалит указанный класс если он присутствует и добавит его если он

отсутствует с анимированным переходом между различными состояниями оформления */

toggleClass(имя_класса, продолжительсть_перехода)

/* Переключится с класса указанного в первом параметре на второй с

анимированным переходом между различными состояниями оформления */

switchClass(класс, новый_класс, продолжительность_перехода)

Пример

```
$(document).ready(function(){

  $(&amp;quot;#but1&amp;quot;).click(function(){
```

```
    $(&quot;.el1&quot;).addClass(&quot;newclass&quot;,1000);

});

$(&quot;#but2&quot;).click(function(){

    $(&quot;.el1&quot;).removeClass(&quot;newclass&quot;,1000);

});

$(&quot;#but3&quot;).click(function(){

    $(&quot;.el1&quot;).toggleClass(&quot;newclass&quot;,1000);

});

$(&quot;#but4&quot;).click(function(){

    $(&quot;.el1&quot;).switchClass(&quot;el1&quot;,&quot;newclass&quot;);

});
```

});

Попробовать в редакторе

jQuery UI классы оформления

jQuery UI содержит набор предопределенных классов для создания оформления сайтов.

Используя готовые классы Вы можете сэкономить время при создании оформления веб-страниц. Использование готовых классов также поможет стандартизировать оформление Вашего сайта.

С помощью настройщика тем Вы можете настроить оформление предопределенных классов на Ваш вкус.

Виджет themeswitcher

Виджет themeswitcher позволяет переключатся между различными темами оформления не покидая страницы.

Это бывает особенно полезно при подборе подходящего класса оформления.

Синтаксис:

// Добавляем ссылку на плагин (данный код должен располагаться в секции body)

<script type="text/javascript"

** src="http://jqueryui.com/themeroller/themeswitchertool/">**

</script>

//Создаем элемент который будет превращен в виджет

** **

/* В тэге script задаем код, который превратит элемент в виджет

   (данный код должен располагаться в head)*/

$("#switcher").themeswitcher();

Пример

```
$(document).ready(function(){

  $(&quot;#switcher&quot;).themeswitcher({

    initialText:&quot;Выбрать тему&quot;,

    buttonPreText:&quot;Тема: &quot;

  });

});
```

Попробовать в редакторе

Оформление виджетов

С помощью класса **.ui-widget** Вы можете оформить внешний вид виджетов. Данный класс устанавливает одинаковый шрифт всем вложенным элементам тем самым стандартизируя вид содержимого виджета.

С помощью класса **.ui-widget-header** Вы можете оформить заголовок, а с помощью **.ui-widget-content**
содержимое виджетов.

Пример

<h2 style="text-align:center" class="ui-widget-header">....

<p style="text-align:center" class="ui-widget-content" >....

Попробовать в редакторе

Добавление сглаженных углов

С помощью предопределенных классов jQuery UI Вы можете сделать углы элементов сглаженными.

Класс

Описание

.ui-corner-all

Позволяет сделать все углы элемента сглаженными.

.ui-corner-tr

Делает сглаженным верхний правый угол элемента.

.ui-corner-tl

Делает сглаженным верхний левый угол элемента.

.ui-corner-top

Делает сглаженными верхний левый и верхний правый углы элемента.

.ui-corner-bl

Делает сглаженным нижний левый угол элемента.

.ui-corner-br

Делает сглаженным нижний правый угол элемента.

.ui-corner-bottom

Делает сглаженными нижний правый и нижний левый углы элемента.

.ui-corner-left

Делает сглаженными верхний левый и нижний левый углы элемента.

.ui-corner-right

Делает сглаженными верхний правый и нижний правый углы элемента.

Пример

<div id="container" class="ui-widget ui-corner-all">....

<div id="container" class="ui-widget ui-corner-top">....

<div id="container" class="ui-widget ui-corner-tr">....

Попробовать в редакторе

jQuery UI иконки

Иконки помогают создать интуитивно понятный пользовательский интерфейс. К примеру Вы можете добавить иконку в форме знака вопроса на кнопку, которая вызывает окно со справочной информацией, или добавить иконку в форме конверта на кнопку отправляющую письмо по электронной почте.

jQuery UI предлагает для использования более 170 "встроенных" иконок.

Для того, чтобы вставить иконку необходимо:

1. Создать элемент, в котором будет отображена иконка;
2. Добавить этому элементу класс *ui-icon*;
3. Добавить элементу класс соответствующий желаемой иконке.

Названия классов иконок в jQuery UI имеют следующий стандартный вид: *.ui-icon-[название]-[дополнительное описание]-[направление]*
.

К примеру, иконка жирной стрелки направленной направо, имеет следующий класс: *.ui-icon-arrowthick-1-e*
(arrowthick - "жирная стрелка", e = east - "восток").

Пример

<span class="ui-icon ui-icon-carat-1-n">

<span class="ui-icon ui-icon-carat-1-ne">

<span class="ui-icon ui-icon-carat-1-e">

<span class="ui-icon ui-icon-carat-1-se">

<span class="ui-icon ui-icon-carat-1-s">

<span class="ui-icon ui-icon-grip-solid-horizontal">

<span class="ui-icon ui-icon-gripsmall-diagonal-se">

<span class="ui-icon ui-icon-grip-diagonal-se">

Попробовать в редакторе

Таблица предопределенных классов

Таблица ниже содержит описание всех остальных существующих в jQuery UI предопределенных классов:

Класс

Описание

.ui-helper-hidden

Применяет display:none к элементу.

.ui-helper-reset

Сбрасывает значения таких свойства как margin, padding, text-decoration и list-style.

.ui-state-default

Данный класс применяется к кнопкам, на которые не был наведен курсор мыши.

.ui-state-hover

Данный класс применяется к кнопкам, на которые был наведен курсор мыши.

.ui-state-active

Данный класс применяется к кнопкам во время щелчка мыши.

.ui-state-highlight

Данный класс применяется к элементам, на которые необходимо обратить внимание.

.ui-state-error

Данный класс применяется к элементам содержащим сообщения об ошибках.

.ui-state-disabled

Данный класс применяется к недоступным для использования элементам.

.ui-priority-primary

С помощью данного класса Вы можете выделить кнопку из группы кнопок как наиболее важн

.ui-priority-secondary

С помощью данного класса Вы можете выделить кнопку из группы кнопок как наименее важ

Пример

<ul class="ui-helper-reset">....

<button id="but1" class="ui-state-default">....

<p class="ui-state-highlight">....

<p class="ui-state-error">....

<button id="but2" class="ui-state-disabled">....

<button class="ui-priority-secondary">....

<button class="ui-priority-primary">....

Попробовать в редакторе

Тэги: на , все , будет , со , будут , элементы , выбраны , странице , input , атрибутом , списка , элемент , атрибута ,

все