Базовые фильтры

- **:first** - выбирает первый элемент соответствующего селектора.

- **$("label:first").css("font-style", "italic");**
- Данная инструкция выделит курсивом первый элемент label

- **:last** - выбирает последний элемент соответствующего селектора

- Пример: $("label:last").css("font-style", "italic");
- выделит курсивом последний элемент label

- **:even** - выбирает четные элементы (отсчет от нуля).
- **$("tr:even").css("font-weight", "bold")**
- Выделит жирным шрифтом четные строки

- **:odd** - выбирает нечетные элементы (отсчет от нуля).
- **$("tr:odd").css("font-style", "italic"); -**
- курсивом все нечетные строки таблицы (т.к. нумерация идет с нуля, то зрительно кажется, что четные строки).

:eq(index) - выбирает элемент по его индексу (начиная с нуля).

$("td:eq(2)").css("color", "red") Выделит красным цветом текст второй ячейки таблицы (т.к. нумерация идет с нуля, то зрительно кажется, что третьей ячейки).

:gt(index) - выбирает элементы с индексом больше указанного.

Пример: **$("td:gt(2)").css("color", "blue");** Выделит красным цветом текст во всех ячейках таблицы, начиная со 2

:lt(index) - выбирает элементы с индексом меньше указанного.

$("td:lt(4)") **.css("color", "green")**; Выделит зеленым текст в первых пяти ячейках таблицы

:header - выбирает все элементы, которые являются заголовками (h1, h2...) $(":header").css("color", "red");

Данная инструкция сделает все заголовки красными

- **not:(selector) Выбирает все элементы, не соответстующие селектору**
- **$("table tr:not(.tr4)").css("background-color","red");**

Селекторы потомков.

:first-child Возвращает: Массив<Элемент(ы)>

Выбирает все элементы, которые являются первыми дочерними объектами у “родителей”.

:last-child Возвращает: Массив<Элемент(ы)>

Выбирает все элементы, которые являются последними дочерними объектами у “родителей”.

:only-child Возвращает: Массив<Элемент(ы)>

Выбирает все элементы, которые являются единственными дочерними объектами у “родителей”.

:nth-child(индекс/even/odd/an+b)

Возвращает: Массив<Элемент(ы)>

Выбирает все элементы, которые являются “N-ными потомками” либо четными/нечетными “потомками”.

```
$("ul.drow li:first-child").css("border","solid 3px red");

$("ul.drow li:last-child").css("border","solid 3px blue");

$("span:only-child").css("color","#335599");

$("ul.drow li:nth-child(even)"). css("background-color","#335599");

$("ul.drow li:nth-child(odd)").css("background-color","#74BD5C");

$("ul.drow li:nth-child(1)").css("border","5px dotted green");
```
(нумерация с 1)
```
$("table tr:nth-child(3n+1)").css("background-color","blue");
```

Фильтры содержимого

:contains (text) - выбирает элементы, которые содержат заданный текст (text).

$("label:contains('Cube')").css("text-decoration", "underline"); - подчеркнет все элементы label, содержащие подстроку 'Cube'.

:**empty** - выбирает все элементы, которые не содержат потомков (т.е. являются пустыми).

Пример: $("td:empty").text("-----");

Данная инструкция найдет все пустые ячейки таблицы и вставит в них текст "-----".

:has(selector) - выбирает элементы, которые содержат хотя бы один элемент, указанный в селекторе.

Пример: $("li:has(div)").css("border", "dotted 2px green"); Данная инструкция найдет те li-, в которых есть хотя бы один div и обведет рамкой

:parent - выбирает родительские элементы, т.е. те, у которых есть потомки.

Пример: $("td:parent").css("font-style", "italic"); Данная инструкция найдет все ячейки таблицы, в которых есть текст или еще что-либо, и сделает их шрифт курсивом.

Расширение набора выбранных элементов

Метод **add()** позволяет добавить в существующий объект jQuery дополнительные элементы.

Варианты вызова метода **add ()**

add (селектор), add (селектор, контекст)-

Добавляет в текущий набор дополнительные элементы, соответствующие селектору, без учета и с учетом контекста

add (HTMLElement) , add(HTMLElement []) Добавляет в текущий набор элемент или массив элементов HTMLElement

add (jQuery) Добавляет в текущий набор содержимое указанного объекта jQuery

```
<script type ="text/javascript">
$(document).ready(function() {
 var labelElems =
document.getElementsByTagName("label");
 var jq= $('img[src*=Cube]');
$('img:even').add(jq).add(labelElems)
.css("border", "thick double red");


});
</script>
```

Сужение набора выбранных элементов

Существует ряд методов, позволяющих удалять элементы из существующего набора. Каждый из этих методов возвращает новый объект jQuery, содержащий урезанный набор элементов. Объект jQuery, для которого вызывается метод, остается неизменным.

eq(индекс) Исключает из набора все элементы, кроме элемента с указанным индексом

first() Исключает из набора все элементы, кроме первого

last() Исключает из набора все элементы, кроме последнего

filter(условие)

Исключает из набора элементы, не соответствующие указанному условию.

has() , has(jQuery), has(HTMLElement), has(HTMLElement[])-

Исключает из набора элементы, у которых отсутствуют потомки, соответствующие указанному селектору или объекту jQuery, или потомки, не включающие указанные Объекты HTMLElement

not (условие) Исключает из набора все элементы, соответствующие указанному условию.

slice(начало,конец)

Исключает из набора все элементы, индексы которых выходят за пределы указанного диапазона

Сужение набора до одного элемента

Простейшими методами с помощью которых можно сократить набор выбранных элементов, являются методы **first (), last () и eq(номер**) – номер отчитывается от нуля. Эти три метода позволяют выбрать конкретный элемент на основании его позиции в наборе элементов, содержащихся в объекте jQuery.

```
<script type="text/javascript">
$(document).ready(function() {
var jq = $('label');
// выбор первого элемента и воздействие на него
jq.first().css("border", "thick double red");
// выбор последнего элемента и воздействие на него
jq.last().css("border", "thick double green");
// выбор элемента с указанным индексом и воздействие  на него
jq.eq(2).css("border", "thick double black");
jq.eq(-2).css("border", "thick double black");
});
</script>
```

Сужение набора до элементов, индексы которых принадлежат к заданному диапазону

Если необходимо оставить в выбранном наборе лишь элементы, индексы которых принадлежат к заданному диапазону, используется метод slice().

В качестве аргументов метод slice() принимает значения начального и конечного индексов.

Отсчет индексов ведется от нуля, причем элемент, которому соответствует конечный индекс, в результирующий набор не включается. Поэтому аргументам 0 и 2 соответствует выбор первых двух элементов. Если второй аргумент опущен, то выбор элементов продолжается до конца существующего набора. Следовательно, использованию единственного аргумента 4 для набора из шести элементов соответствует выбор последних двух элементов (с индексами 4 и 5).

```
var jq = $('label');
jq.slice(0, 2).css("border", "thick double black");
jq.slice(4).css("border", "thick solid red");
```

Фильтрация элементов

Метод **filter()** позволяет задать условие. Элементы, не удовлетворяющие заданному условию, исключаются из набора.

Вариант вызова

- **filter(селектор)** Исключает из набора элементы, которые не соответствуют указанному селектору
- **filter(HTMLElement)** Исключает из набора все элементы, кроме указанного
- **flter(jQuery)** Исключает из набора все элементы, которые не содержатся в указанном Объекте jQuery
- **filter(функция(индекс))** Указанная функция вызывается для каждого элемента набора; из набора исключаются все элементы, для которых функция возвращает значение false

```
// удаление элементов, в значении атрибута src которых  нет  'Cube'
$('img').filter("[src*=Cube]").css("border", "thick double red");
```

// удаление элементов, не содержащих 'pr_1'

```
 var jq = $('[for*=pr_1]');
$('label').filter(jq) .css("color", "blue");
```

// удаление элементов, не являющихся указанным элементом

```
var elem =
    document.getElementsByTagName("label")[2];
$('label').filter(elem).css("font-size", "1.5em");
```

// удаление элементов с использованием функций

```
$('img').filter(function(index) {
 return (index == 4);
}).css("border", "thick solid red");
```

ПредоставляемаяjQuery функция выполняется по одному разу для каждого элемента набора, содержащегося в объекте jQuery. Если эта функция возвращает true, то элемент, для которого она была вызвана, остается в наборе. Если же возвращаемым значением является false то он исключается

```
$( "li" )
.filter(function( index ) {
return index % 3 === 2;
})
.css( "background-color", "red" );
```

Метод not () работает аналогично методу filter (), но обращает процесс фильтрации.

Вариант вызова

not (селектор) Удаляет из выбранного набора элементы, соответствующие селектору

not(HTMLElement),not (HTMLElement []) Удаляет из выбранного набора указанный элемент или элементы

not (jQuery) Удаляет из выбранного набора элементы, которые содержатся в указанном объекте jQuery

not (функция (индекс))

Указанная функция вызывается для каждого элемента набора; из набора исключаются все элементы, для которых функция возвращает значение true

Сужение набора до элементов, имеющих определенных потомков

Метод has () можно использовать для того, чтобы оставить в наборе выбранных элементов только те из них, у которых есть определенные потомки, указываемые либо с помощью селектора, либо с помощью одного или нескольких объектов HTMLElement. Пример использования метода has().

```
// использование метода has
$('li.dcell').has('img[src*=Cube]').css("border", "thick solid red");
var jq = $('[for*=pr_1]');
$('li.dcell').has(jq).css("border", "thick solid blue");
```

Преобразование набора выбранных элементов

Метод map () обеспечивает гибкий способ использования одного объекта jQuery для создания другого.

В качестве аргумента методу map () передается функция. Эта функция вызывается для каждого из элементов, входящих в исходный объект jQuery, а возвращаемые ею объекты HTMLElement включаются в результирующий объект jQuery .

```
$('li.dcell').map(function(index,elem) {
return elem.getElementsByTagName("img")[0];
}).css("border", "thick solid red");

$('img').map(function(index,elem) {
var newsrc=$(elem).attr('src').toLowerCase();
$(elem).attr('src',newsrc);
console.log($(elem).attr('src'));
return elem;
}).css("border", "thick dotted yellow");
```

Контроль набора выбранных элементов

Чтобы выяснить, соответствует ли хотя бы один из выбранных элементов заданному условию, можно использовать метод is().

is(селектор) Возвращает true, если объект jQuery содержит хотя бы один элемент, соответствующий селектору

is(HTMLElement), is(HTMLElement[]) Возвращает true, если объект jQuery содержит указанный элемент или хотя бы один из элементов указанного массива

is(jQuery) Возвращает true, если объект jQuery содержит хотя бы один из элементов, которые содержатся в объекте, переданном в качестве аргумента

is(функция(индекс))

Возвращает true, если функция возвращает true хотя бы один раз

```
var isResult = $('img').is(function(index) {
return index == 2
});
console.log("Результат: " + isResult);
```

Навигация по дереву DOM

Выбранный набор элементов можно использовать в качестве отправной точки для перемещения к другим узлам DOM-дерева.

При этом, один набор элементов используется для создания другого набора.

Каждый из описанных далее методов возвращает объект jQuery. Этот объект может как содержать подходящие объекты, если таковые имеются, так и быть пустым в случае их отсутствия (свойство length таких объектов возвращает нулевое значение).

Перемещение вниз по дереву

Процесс перемещения вниз по иерархической структуре DOM связан с выбором **дочерних** элементов (непосредственных потомков), а также всех **остальных элементов**, являющихся потомками элементов, содержащихся в объекте jQuery.

Методы, используемые для перемещения вниз по иерархической структуре DOM

children() Выбирает дочерние элементы всех элементов, содержащихся в объекте jQuery

children(селектор) Выбирает все элементы, которые соответствуют указанному селектору и при этом являются непосредственными потомками элементов, содержащихся В Объекте **jQuery**

contents() Возвращает дочерние элементы и текстовое содержимое всех элементов, содержащихся в объекте jQuery

find(селектор)

Выбирает элементы, которые соответствуют указанному селектору и при этом являются потомками элементов, содержащихся в объекте jQuery

find(jQuery) ,find(HTMLElement),find(HTMLElement[]) Выбирает пересечение множества непосредственных потомков элементов, содержащихся в объекте jQuery, и множества элементов, содержащихся в объекте аргумента

Метод **children()** выбирает лишь те элементы, которые являются непосредственными потомками (дочерними элементами) элементов, содержащихся в объекте jQuery, и может принимать селектор в качестве необязательного аргумента, обеспечивающего дополнительную фильтрацию элементов. Метод **find()** предназначен для выбора всех потомков, а не только дочерних элементов. Метод contents () наряду с дочерними элементами возвращает также текстовое содержимое.

```
var allchilds=$('ul.drow').children();
var childCount = allchilds.length;
allchilds.each(function(index, elem) {
console.log("Дочерний элемент: " + elem.tagName + " " +
    elem.className+" "+ index);
});
console.log("Всего имеется " + childCount +
" дочерних элементов");
```

Потомки

```
var allDescandant=$('ul.drow').find('*'); // все
    потомки
var descCount = allDescandant.length;
allDescandant.each(function(index, elem) {
console.log("Потомок элемент: " + elem.tagName
    + " " + elem.className+" "+ index);
});
console.log("Всего имеется " + descCount +
"  потомков");
```

```
<div class="answer">
<p><span>Hello</span>, how are you?</p>
<p>Me? I'm <span>good</span>.</p>
</div>
<script>
$( "div.answer" ).find( "span" ).css( "color",
   "red" );
</script>
```

Перемещение вверх по дереву

Перемещению вверх по DOM-дереву соответствует поиск родителей и предков элементов, содержащихся в объекте jQuery. Методы, используемые для таких перемещений, приведены в табл.

Метод	Описание
`closest(`*селектор*`)` `closest(`*селектор, контекст*`)`	Выбор ближайшего предка, соответствующего указанному селектору, для каждого элемента, содержащегося в объекте `jQuery`
`closest(jQuery)` `closest(HTMLElement)`	Выбор ближайшего предка для каждого элемента в объекте `jQuery`, совпадающего с одним из элементов, содержащихся в объекте аргумента
`offsetParent()`	Нахождение ближайшего предка, значением CSS-свойства `position` которого является `fixed`, `absolute` или `relative`
`parent()` `parent(`*селектор*`)`	Выбор непосредственных предков для каждого элемента в объекте `jQuery` с возможностью их фильтрации с помощью селектора
`parents()` `parents(`*селектор*`)`	Выбор предков для каждого элемента в объекте `jQuery` с возможностью их фильтрации с помощью селектора

parentsUntii (HTMLElement) parentsUntil (HTMLElement, селектор),
parentsUntii (HTMLElement []),
parentsUntil (HTMLElement [] , селектор)
Выбирает предков для каждого элемента в объекте jQuery ДО тех пор,пока не встретится один из указанных элементов. Результаты могут фильтроватьСЯ посредством второго селектора

```
// выбор родителей
$('li.dcell').parent().each(function(index, elem) { console.log("Элемент: " + elem.tagName + " " + elem.id);
});
// выбор родителей с фильрацией
$('li.dcell').parent('#row1')
.each(function(index, elem) { console.log("0тфильтрованный элемент: " + elem.tagName + " " + elem.id);
});
```

Выбор предков

Метод **parents()** обеспечивает возможность выбора всех, а не только непосредственных предков (родителей) элементов, содержащихся в объекте jQuery. Как и в предыдущем случае, метод может принимать в качестве аргумента селектор для фильтрации результатов.

```
$('img[src*=Cube]').parents().each(function(index, elem) {console.log("Элемент: " + elem.tagName + " " + elem.className + " " + index);
});
```

Метод **parentsUntil ()** является еще одной разновидностью методов, предназначенных для выбора предков элементов. Для каждого из элементов, содержащихся в объекте jQuery, метод parentsUntil () осуществляет перемещение вверх по иерархической структуре DOM, выбирая элементы-предки до тех пор, пока не встретится элемент, соответствующий селектору. Пример использования этого метода

```
$('label[for=pr_1]').parentsUntil("body").each(function(index
    , elem) {console.log("Элемент: " + elem.tagName + " " +
    elem.className + " " + index);
});
```

При этом элементы, соответствующие селектору, исключаются из состава выбираемых предков. В данном случае это означает исключение элемента body

Выбор первого подходящего предка

Метод closest () позволяет выбирать первого из предков, соответствующих селектору, для каждого элемента в объекте jQuery. Пример использования этого метод

```
$('img').closest('.drow').each(function(index,elem) {
  console.log("Элемент: " + elem.tagName +
" " + elem.className + " " + elem.id);
});
```