Глава 2

Селекторы

Как вы уже знаете, в качестве параметра функции `$()` могут быть указаны селекторы, которые являются частью стандарта CSS. Рассмотрим различные возможные селекторы подробно.

2.1. Часто используемые селекторы

Перечислим наиболее часто используемые селекторы.

- `*` — коллекция всех тегов

  ```
  var n = $("*").size(); // Количество тегов
  ```

- *`Тег`* — коллекция всех тегов, имеющих указанное имя

  ```
  $("p").css("backgroundColor", "red");
  // Делаем фон всех абзацев красным
  ```

- #*`Идентификатор`* — ссылка на элемент с указанным идентификатором

  ```
  $("#id1").css("backgroundColor", "red");
  // Делаем фон красным для элемента с id="id1"
  ```

  (Если элементов с одинаковым идентификатором несколько, то будет возвращена ссылка только на первый элемент)

- *`Тег`*#*`Идентификатор`* — ссылка на элемент с указанным идентификатором, который расположен в определенном теге

  ```
  $("p#id1").css("backgroundColor", "red");
  // Делаем фон красным для абзаца с id="id1"
  ```

  (Если абзацев с одинаковым идентификатором несколько, то будет возвращена ссылка только на первый абзац. Если идентификатор находится в другом теге, то он будет проигнорирован)

- .*`Класс`* — коллекция элементов, имеющих указанный класс

  ```
  $(".cls2").css("backgroundColor", "red");
  // Делаем фон красным для всех элементов с class="cls2"
  ```

- *`Тег`*.*`Класс`* — коллекция элементов, имеющих указанный класс в определенном теге

  ```
  $("p.cls2").css("backgroundColor", "red");
  // Делаем фон красным для всех абзацев с class="cls2"
  ```

Примечание

Для ускорения выборки не следует указывать название тега перед идентификатором. Например, пишите `"#id1"` вместо `"p#id1"`. Если поиск производится по стилевому классу, то, наоборот, следует указать название тега. Например, пишите `"p.cls1"` вместо `".cls1"`.

Если название идентификатора (или класса) содержит специальные символы (например, точку или квадратные скобки), то их необходимо экранировать двумя слешами (\\).

```
$("#div1\\.index\\[5\\]").html("Текст");
```

В этом примере получаем ссылку на следующий элемент.

```
<div id="div1.index[5]"></div>
```

2.2. Группирование селекторов

Если необходимо, например, применить один стиль к разным элементам, то можно указать селекторы через запятую.

```
$("#id1, div").addClass("newClass");
```

В этом примере для элемента с идентификатором `id1`, а также для всех тегов `<div>` назначается стилевой класс `newClass`.

2.3. Привязка к элементам документа

Выполнить поиск элемента с учетом заданного отношения к другому элементу HTML-документа можно следующими способами.

- *Элемент1 Элемент2* — находим *Элемент2*, который располагается внутри контейнера *Элемента1*

  ```
  $("div a").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если тег `<a>` находится внутри тега `<div>`.

  ```
  <div><a href="link.html">Ссылка</a></div>
  ```

- *Элемент1 > Элемент2* — находим *Элемент2*, который является дочерним для *Элемента1*

  ```
  $("div > a").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если тег `<a>` находится внутри тега `<div>` и не вложен в другой тег.

  ```
  <div>
  <a href="link1.html">Ссылка 1</a><br>
  <span><a href="link2.html">Ссылка 2</a></span>
  </div>
  ```

  В этом примере только первая ссылка станет красного цвета, так как вторая ссылка расположена внутри тега `<span>`.

- *Элемент1 + Элемент2* — находим *Элемент2*, который является соседним для *Элемента1* и следует сразу после него.

  ```
  $("div + a").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если тег `<a>` следует сразу после тега `<div>`.

  ```
  <div>Текст</div><a href="link.html">Ссылка</a>
  ```

- *Элемент1 ~ Элемент2* — находим *Элемент2*, который следует после *Элемента1*, причем необязательно сразу.

  ```
  $("div ~ a").css("color", "red");
  ```

Цвет текста ссылки станет красным, если тег <a> следует после тега .

```
<div>Текст</div>
<span>Текст</span><br>
<a href="link1.html">Ссылка 1</a><br>
<a href="link2.html">Ссылка 2</a><br>
<span><a href="link3.html">Ссылка 3</a></span><br>
<a href="link4.html">Ссылка 4</a><br>
```

В этом примере ссылки 1, 2 и 4 станут красного цвета. Ссылка 3 не станет красного цвета, так как расположена внутри тега .

При необходимости можно составлять выражения из нескольких селекторов

```
$("div span a").css("color", "red");
```

Цвет текста ссылки станет красным, если тег <a> расположен внутри тега , а тот в свою очередь вложен в тег

```
<div>
<a href="link1.html">Ссылка 1</a><br>
<span>
<a href="link2.html">Ссылка 2</a><br>
</span>
</div>
```

В этом примере только ссылка 2 будет красного цвета.

2.4. Привязка к параметрам тегов

Для привязки к параметрам тегов применяются следующие селекторы.

- [*Параметр*] — элементы с указанным параметром

  ```
  $("a[id]").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если тег <a> имеет параметр id.

  ```
  <a id="link1" href="link1.html">Ссылка 1</a>
  ```

- [*Параметр*='*Значение*'] — коллекция элементов, у которых параметр точно равен значению

  ```
  $("a[href='link1.html']").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если параметр href тега <a> имеет значение "link1.html".

- [*Параметр*!='*Значение*'] — коллекция элементов, у которых параметр не равен значению

  ```
  $("a[href!='link1.html']").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если параметр href тега <a> не имеет значение "link1.html".

- [*Параметр*^='*Значение*'] — коллекция элементов, у которых параметр начинается с указанного значения

  ```
  $("a[href^='li']").css("color", "red");
  ```

Цвет текста ссылки станет красным, если значение параметра `href` тега `<a>` начинается с `"li"`.

- `[`*Параметр*`$='`*Значение*`']` — коллекция элементов, у которых параметр оканчивается указанным значением

  ```
  $("a[href$='.html']").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если значение параметра `href` тега `<a>` оканчивается на расширение `".html"`.

- `[`*Параметр*`*='`*Значение*`']` — коллекция элементов, у которых параметр содержит указанный фрагмент значения

  ```
  $("a[href*='link']").css("color", "red");
  ```

  Цвет текста ссылки станет красным, если значение параметра `href` тега `<a>` содержит фрагмент `"link"`.

Если необходимо сделать привязку сразу к нескольким параметрам, то используется следующий формат.

`[`*Параметр*`='`*Значение*`'][`*Параметр*`='`*Значение*`']`

Пример:

```
$("a[href='link1.html'][id*='link']").css("color", "red");
```

Цвет текста ссылки станет красным, если значение параметра `href` тега `<a>` равно `"link1.html"`, а параметр `id` содержит фрагмент `"link"`.

2.5. Псевдоклассы

В библиотеке jQuery используются следующие псевдоклассы, определенные в CSS.

- `:nth-child(`*N*`)` — элемент, являющийся *N*-м дочерним элементом своего родительского элемента. Нумерация элементов начинается с 1. В качестве параметра можно указать индекс или два специальных слова:

  ◊ `even` — все четные элементы;

  ◊ `odd` — все нечетные элементы.

  Выделим все нечетные пункты списка шрифтом красного цвета.

  ```
  $("ul li:nth-child(odd)").css("color", "red");
  ```

  В качестве параметра можно также указывать следующее значение.

  `<`*Сколько элементов*`>n+<`*Начальная позиция*`>`

  Выделим каждый третий пункт списка, начиная со второго.

  ```
  $("ul li:nth-child(3n+2)").css("color", "red");
  ```

  В итоге будут выделены пункты 2-й, 5-й, 8-й и т.д.

- `:first-child` — элемент, являющийся первым дочерним элементом своего родительского элемента. Выделим первый пункт списка шрифтом красного цвета.

  ```
  $("ul li:first-child").css("color", "red");
  ```

- `:last-child` — элемент, являющийся последним дочерним элементом своего родительского элемента. Выделим последний пункт списка шрифтом красного цвета.

  ```
  $("ul li:last-child").css("color", "red");
  ```

- `:only-child` — элемент, являющийся единственным дочерним элементом своего родительского элемента.

  ```
  $("ul li:only-child").css("color", "red");
  ```

  Пункт списка будет выведет шрифтом красного цвета, если других пунктов нет.

- `:empty` — пустой элемент.

  ```
  $("div:empty").text("Пустой DIV");
  ```

  Во всех пустых тегах `<div>` выводим текст `"Пустой DIV"`.

  ```
  <div>Текст</div>
  <div>
  </div>
  <div></div>
  ```

  В этом примере текст будет выведен только в последнем теге `<div>`.

- `:not(`*Селектор*`)` — элемент, не соответствующий селектору.

  ```
  $("a:not(#link1)").css("color", "red");
  ```

  Все ссылки в документе станут красного цвета, за исключением элемента с идентификатором `link1`.

- `:has(`*Селектор*`)` — все элементы, содержащие хотя бы один элемент, соответствующий селектору.

  ```
  $("div:has(a)").css("background-color", "red");
  ```

  Выделяем все теги `<div>`, в которых содержится хотя бы одна ссылка.

Для привязки к элементам формы предназначены следующие псевдоклассы.

- `:input` — все элементы формы (теги `<input>`, `<select>`, `<textarea>`, `<button>`).
- `:text` — все текстовые поля (`type="text"`).
- `:password` — все поля для ввода паролей (`type="password"`).
- `:radio` — все переключатели (`type="radio"`).
- `:checkbox` — все поля для установки флажков (`type="checkbox"`).
- `:submit` — все кнопки для отправки формы (`type="submit"`).
- `:reset` — все кнопки очистки формы (`type="reset"`).
- `:button` — все обычные кнопки (`type="button"`, а также теги `<button>`).
- `:file` — все поля для отправки файлов (`type="file"`).
- `:enabled` — активный элемент.
- `:disabled` — неактивный элемент. В качестве примера выведем в активном текстовом поле текст Активный элемент, а в неактивном — текст Неактивный элемент.

```
$(document).ready(function() {
   $("#txt1")[0].disabled = true;
   $(":text:enabled").val("Активный элемент");
   $(":text:disabled").val("Неактивный элемент");
});

<input id="txt1" type="text"><br>
<input id="txt2" type="text">
```

После формирования структуры документа делаем неактивным элемент с идентификатором `txt1`. Затем, в зависимости от активности элемента, выводим соответствующее сообщение с помощью метода `val()`.

- `:checked` — отмеченный элемент (например, установленный флажок). Выведем количество установленных флажков.

```
alert($(":checkbox:checked").length);
```

- `:selected` — все выбранные элементы списка (тег `<select>`). Выведем текст пункта списка после его выбора.

```
$(document).ready(function() {
   $("#sel1").change(function() {
      alert($("#sel1 option:selected").text());
   });
});

<select id="sel1">
<option>Пункт 1</option>
<option>Пункт 2</option>
<option>Пункт 3</option>
<option>Пункт 4</option>
</select>
```

Рассмотрим другие псевдоклассы.

- `:even` — все четные элементы (элементы нумеруются с 0).
- `:odd` — все нечетные элементы (элементы нумеруются с 0). В качестве примера назначим разные стилевые классы четным и нечетным строкам таблицы.

```
$("table tr:even").addClass("clsEven");
$("table tr:odd").addClass("clsOdd");
```

- `:eq(`*Индекс*`)` и `:nth(`*Индекс*`)` — элемент с указанным индексом (элементы нумеруются с 0). Выделим третий пункт списка.

```
$("ul li:eq(2)").css("color", "red");
```

Выделим первый пункт списка.

```
$("ul li:nth(0)").css("color", "red");
```

- `:gt(`*Индекс*`)` — элементы с индексом, большим, чем указан (элементы нумеруются с 0). Выделим все пункты списка, кроме первого.

```
$("ul li:gt(0)").css("color", "red");
```

- `:lt(`*Индекс*`)` — элементы с индексом, меньшим, чем указан (элементы нумеруются с 0). Выделим первые два пункта списка.

```
$("ul li:lt(2)").css("color", "red");
```

- `:first` — первый элемент.
- `:last` — последний элемент.
- `:parent` — все элементы, у которых есть дочерние элементы (включая текст).
- `:contains('`*Текст*`')` — все элементы, которые содержат указанный текст в любом месте. Выделим пункты списка, которые содержат фрагмент `"Пункт 1"`.

  ```
  $("ul li:contains('Пункт 1')").css("color", "red");
  ```

  В этом случае будут выделены пункты, содержащие также текст `"Пункт 11"`, `"Пункт 12"` и т.д.
- `:visible` — все видимые элементы (элементы с атрибутом `display` равным `block` или `inline`, а также элементы с атрибутом `visibility` равным `visible`).
- `:hidden` — все невидимые элементы (элементы с атрибутом `display` равным `none`, элементы с атрибутом `visibility` равным `hidden`, а также скрытые элементы форм (`type="hidden"`)).
- `:header` — все заголовки (теги `<h1>...<h6>`). Выделим все заголовки шрифтом белого цвета на черном фоне.

  ```
  $(":header").css({ backgroundColor:"black", color:"white" });
  ```
- `:animated` — все элементы, с которыми выполняется анимация. Прервем все анимации.

  ```
  $(":animated").stop(true, true);
  ```