Геннадий Самков

jQuery
Сборник рецептов

Санкт-Петербург
«БХВ-Петербург»
2010

УДК 681.3.068+800.92 jQuery
ББК 32.973.26-018.1
С17

Самков Г. А.

С17 jQuery. Сборник рецептов. — СПб.: БХВ-Петербург, 2010. — 416 с.: ил. + CD-ROM — (Профессиональное программирование)

ISBN 978-5-9775-0495-9

Книга является сборником решений наиболее часто встречающихся задач при веб-программировании пользовательских интерфейсов с использованием библиотеки jQuery. Рассмотрены практически все методы и вспомогательные функции jQuery, в том числе обеспечивающие взаимодействие jQuery и AJAX. Подробно рассказано о надстройке UI jQuery и приведены описания всех настроек для виджетов, входящих в ее состав, что позволяет использовать книгу в качестве справочника. Приведено большое количество примеров использования наиболее популярных плагинов для jQuery — создание графиков и диаграмм, фотогалерей, навигационных меню, всплывающих подсказок, работа с веб-формами, таймерами и cookies, обработка табличных данных. Компакт-диск содержит примеры, разобранные в книге, файлы библиотеки jQuery версий 1.2.6 и 1.3.2, файлы надстройки UI jQuery, а также файлы рассмотренных в книге расширений сторонних разработчиков.

Для веб-программистов

УДК 681.3.068+800.92 jQuery
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Леонид Кочин*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 03.12.09.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 33,54.
Тираж 1500 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0495-9

Оглавление

Введение

Предлагаемая книга является сборником решений наиболее часто встречающихся задач при программировании пользовательских интерфейсов с использованием JavaScript-библиотеки jQuery.

Предполагается, что читатель знаком с основами JavaScript, CSS и HTML. Кроме того, потребуются некоторые знания основ PHP — языка программирования серверных сценариев, который необходим в некоторых примерах, посвященных организации взаимодействия клиент — сервер с применением технологии AJAX.

Книга может служить не только учебником, но и справочником по библиотеке jQuery и надстройке UI jQuery.

Структура книги

Книга содержит две части и приложение.

В *части I* решения задач представлены так, чтобы помочь читателю на простых примерах освоить подавляющее большинство методов библиотеки jQuery, имеющихся в распоряжении разработчика. Подробно освещены такие вопросы, как:

- выбор и фильтрация элементов;
- работа с атрибутами элементов;
- создание визуальных эффектов, в том числе с использованием возможностей надстройки UI jQuery;
- работа с CSS-свойствами элементов;
- организация доступа к свойствам объекта jQuery;
- манипуляции элементами: изменение содержимого, добавление, замена, удаление и копирование элементов;

- перемещение по элементам объектной модели документа;
- работа с событиями (назначение, удаление и вызов событий);
- взаимодействие jQuery и AJAX;
- некоторые полезные утилиты библиотеки jQuery.

В *части II* приведены решения на основе наиболее популярных расширений для библиотеки jQuery, в том числе рассмотрен официальный пакет расширений UI jQuery. Подробно рассматриваются:

- меню для веб-сайтов (многоуровневые меню, меню на основе плагина TreeView, а также меню, похожее на популярный виджет Accordion);
- организация работы с данными, представленными в табличной форме, рассматривается на примере плагина DataTables;
- возможности реализации графиков и диаграмм на страницах веб-сайта демонстрируются на примере плагина jqPlot;
- работа с AJAX-формами — плагины Autocomplete (автоматические подсказки в форме), Form (организация AJAX-формы), Validate (проверка данных в AJAX-форме) и FileUpload (загрузка файлов на сервер);
- фотогалереи для веб-сайтов рассматриваются на примере плагина FancyBox, кроме того, приведено решение по созданию простейшей встроенной фотогалереи из нескольких строк jQuery-кода;
- некоторые полезные плагины — плагин jQuery Corner (уголки без использования графики), плагин jQuery Cookie (установка и считывание cookie), плагин jQuery ClueTip (всплывающие подсказки), плагин jQuery Timers (управление таймерами);
- виджеты надстройки UI jQuery — Accordion (раскрывающееся меню), Datepicker (выбор даты), Dialog — (диалоговое окно), ProgressBar (шкала загрузки), Slider (шкала с бегунком) и Tabs (организация переключения вкладок);
- надстройка UI jQuery — взаимодействие с элементами страницы: Draggable (перемещение элементов), Droppable ("сброс" элементов), Resizable (изменение размеров элементов), Selectable (выбор элементов), Sortable (сортировка элементов).

В *приложении* описан компакт-диск, прилагаемый к книге.

Как работать с книгой

Книга в основном ориентирована на разработчика, располагающего компьютером с операционной системой Windows, но пользователь UNIX также сможет выполнить на своем компьютере все примеры.

В ходе чтения следует выполнять на своем компьютере все примеры, описываемые в книге. Рекомендуем читателю самостоятельно изменять и переделывать каждый пример, чтобы лучше понять, как он работает.

Автор приложил все усилия, чтобы изложить материал с наибольшей точностью, но не исключает возможности ошибок и опечаток. Автор также не несет ответственности за последствия использования сведений, изложенных в книге.

Источники информации

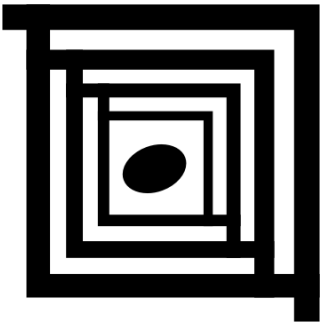
В книге невозможно охватить все вопросы, и читателю наверняка потребуются дополнительные сведения, например, из сети Интернет. Кроме того, могут изменяться версии программного обеспечения, рассматриваемого в книге. Вот адреса, которыми вы можете воспользоваться:

- **http://jquery.com** — официальный сайт библиотеки jQuery (англ.);
- **http://docs.jquery.com/Main_Page** — оригинальная документация по библиотеке jQuery (англ.);
- **http://slyweb.ru/jquerymain/** — перевод документации jQuery (рус.);
- **http://jquery-docs.ru** — перевод документации jQuery (рус.);
- **http://jqueryui.com** — официальный сайт надстройки UI jQuery (англ.);
- **http://www.linkexchanger.su** — блог автора книги, содержит много статей с примерами использования библиотеки jQuery (рус.);
- **http://plugins.jquery.com/** — на сайте представлено большое количество разнообразных плагинов для библиотеки jQuery (англ.);
- **http://bassistance.de/jquery-plugins/jquery-plugin-treeview/** — страница плагина jQury TreeView (англ.);
- **http://www.datatables.net** — сайт плагина jQuery DataTables (англ.);
- **http://www.jqplot.com** — сайт плагина jqPlot (англ.);
- **http://bassistance.de/jquery-plugins/jquery-plugin-autocomplete/** — страница плагина jQury Autocomplete (англ.);
- **http://malsup.com/jquery/form/** — сайт плагина jQuery Form (англ.);
- **http://bassistance.de/jquery-plugins/jquery-plugin-validation/** — страница плагина jQury Validation (англ);
- **http://www.uploadify.com** — сайт плагина jQuery Uploadify (англ.);
- **http://fancybox.net** — сайт плагина jQuery FancyBox (англ.);

- **http://plugins.learningjquery.com/cluetip/** — страница плагина jQuery ClueTip (англ.);
- **http://www.stilbuero.de/2006/09/17/cookie-plugin-for-jquery/** — страница плагина jQuery Cookie (англ.);
- **http://jquery.malsup.com/corner/** — страница плагина jQuery Corner (англ.);
- **http://jquery.offput.ca/timers/** — страница плагина jQuuery Timers (англ.);
- **http://firebug.ru** — сайт, посвященный Firebug, замечательному средству отладки JavaScript-кода (рус.).

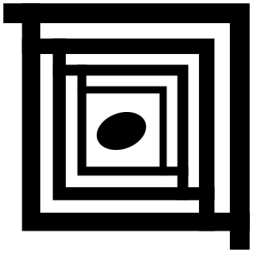
Благодарности

Автор приносит свою благодарность преподавателю компьютерных дисциплин Санкт-Петербургского государственного политехнического университета Елене Сергеевне Бенкен за помощь и поддержку.



ЧАСТЬ I

Методы библиотеки jQuery



ГЛАВА 1

Выбор элементов

А для чего вообще нужно выбирать элементы? Ответ простой — для того, чтобы как-то на них воздействовать. Например, можно изменить атрибуты или CSS-свойства элементов, меняя, таким образом, их визуальное представление, изменять содержимое этих элементов, связывать с ними определенные события и т. д.

Можно выбрать как один элемент, так и множество элементов. Независимо от того, сколько именно элементов будет выбрано, мы будем рассматривать это как набор элементов, называя его объектом jQuery.

1.1. Базовые правила

Проблема

Необходимо отыскать абсолютно все элементы веб-страницы.

Решение

Используем селектор * для решения этой задачи (листинг 1.1.1).

Листинг 1.1.1. Использование селектора *

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-1-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  alert($("*").length);
});
</script>
</head>
<body>
<ul>
  <li></li>
  <li></li>
</ul>
<p></p>
<div><span></span></div>
</body>
</html>
```

Обсуждение

Чтобы рассмотренный пример не выглядел совсем скучно, и можно было понять, что он действительно работает, мы не только выбрали в объект jQuery все элементы веб-страницы, но также подсчитали их число и вывели его в окне предупреждения. Поскольку контекстом в приведенном примере является объект `document`, то в набор попадут элементы не только из `body`, но и из `head`. В наборе также окажутся элементы `script` и т. п.

Проблема

Необходимо отыскать все элементы веб-страницы, но только в контексте `body`, исключив элементы, входящие в `head`.

Решение

Для решения этой задачи так же воспользуемся селектором `*`, но в качестве второго аргумента явно передадим контекст (листинг 1.1.2).

Листинг 1.1.2. Использование селектора *

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
```

```
<title>example-1-1-2</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  alert($("*",document.body).length);
});
</script>
</head>
<body>
<ul>
  <li></li>
  <li></li>
</ul>
<p></p>
<div><span></span></div>
</body>
</html>
```

Обсуждение

Мы точно так же вывели в окно предупреждения число выбранных элементов. Заметили разницу? 6 против 14 в примере из листинга 1.1.1. В набор не попали элементы из `head`.

Проблема

Необходимо отыскать элемент по известному значению его атрибута `id`.

Решение

Для поиска элемента по значению его атрибута `id` воспользуемся селектором идентификатора (листинг 1.1.3).

Листинг 1.1.3. Использование селектора `#id`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-1-3</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
```

```
<script type="text/javascript">
$(function(){
  $("#myDiv").css("border","1px solid #f00");
});
</script>
<style type="text/css">
div { width:150px; height:150px; border:1px solid #00f; margin:2px; }
</style>
</head>
<body>
<div class="myDiv"></div>
<div></div>
<div id="myDiv"></div>
<div><span></span></div>
<div id="otherDiv"></div>
</body>
</html>
```

Обсуждение

Отыскав элемент, который имеет значение идентификатора `myId`, мы применили к нему метод `css()`, добавив выбранному элементу красную рамку шириной в 1 px, чтобы убедиться в том, что селектор действительно обнаружил нужный элемент.

Проблема

Необходимо отыскать элемент по значению атрибута `id`, в который входят специфические символы, такие как точка или квадратные скобки. Проблема состоит в том, что эти символы имеют специальное значение в CSS.

Решение

Снова воспользуемся селектором идентификатора, но перед специальными символами поставим два обратных слэша подряд (листинг 1.1.4).

Листинг 1.1.4. Использование селектора `#id`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-1-4</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $("#my\\.Div").css("border","1px solid #f00");
  $("#my\\[Div\\]").css("border","1px solid #0f0");
});
</script>
<style type="text/css">
div { width:150px; height:150px; border:1px solid #00f; margin:2px; }
</style>
</head>
<body>
<div></div>
<div id="my.Div"></div>
<div><span></span></div>
<div id="my[Div]"></div>
</body>
</html>
```

Обсуждение

Чтобы убедиться, что этот прием работает корректно, мы с помощью метода `css()` устанавливаем для найденных элементов различный цвет рамок.

Проблема

Необходимо отыскать все элементы определенного типа, например, все элементы `div` на веб-странице.

Решение

Для решения задачи нужно всего лишь воспользоваться селектором `element` (листинг 1.1.5).

Листинг 1.1.5. Использование селектора `element`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-1-5</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $("div").css("border","1px solid #f00");
});
</script>
<style type="text/css">
div { width:150px; height:150px; border:1px solid #00f; margin:2px; }
p { width:150px; height:150px; border:1px solid #00f; margin:2px; }
</style>
</head>
<body>
<div></div>
<p></p>
<div></div>
<p></p>
<div></div>
</body>
</html>
```

Обсуждение

С помощью селектора `element` нам удалось выбрать все элементы `div`, имеющиеся на веб-странице. Используя метод `css()`, мы установили для выбранных элементов рамки красного цвета шириной в 1 px.

Проблема

Необходимо отыскать элемент (или элементы) по имени класса.

Решение

Для решения задачи применяется селектор, который в точности повторяет синтаксис `css` (листинг 1.1.6).

Листинг 1.1.6. Использование селектора `.class`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-1-6</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $(".test").css("border","1px solid #f00");
});
</script>
<style type="text/css">
div, p, ul { border:1px solid #00f; margin:2px; }
</style>
</head>
<body>
<div>div</div>
<p class="test">p class="test"</p>
<ul class="test">
  <li>li списка ul class="test"</li>
  <li>li списка ul class="test"</li>
</ul>
<p>p</p>
<div class="test">div class="test"</div>
</body>
</html>
```

Обсуждение

С помощью селектора `.class` мы выбрали все элементы, которые имеют значение `test` атрибута `class`. Для наглядности вновь используем метод `css()`, чтобы установить для выбранных элементов рамки красного цвета шириной в 1 px.

Проблема

Необходимо отыскать элементы, которые имеют сразу несколько имен классов.

Решение

Для решения задачи применяется селектор `.class.class` (листинг 1.1.7).

Листинг 1.1.7. Использование селектора `.class.class`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
```

```
<head>
<title>example-1-1-7</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $(".oneTest.twoTest").css("border","1px solid #f00");
});
</script>
<style type="text/css">
div { border:1px solid #00f; margin:2px; }
</style>
</head>
<body>
<div class="oneTest">div class="oneTest"</div>
<div class="twoTest">div class="twoTest"</div>
<div class="oneTest twoTest">div class="oneTest twoTest"</div>
</body>
</html>
```

Обсуждение

С помощью селектора `.class.class` мы выбрали элемент, который имеет сразу оба названия класса в атрибуте `class`. Опять используем метод `css()`, чтобы установить для выбранных элементов рамки красного цвета шириной в 1 px.

1.2. Выбор элементов с учетом иерархии

Проблема

Необходимо отыскать элементы, являющиеся потомками какого-либо элемента.

Решение

Используем селектор `ancestor descendant` для решения этой задачи (листинг 1.2.1).

Листинг 1.2.1. Использование селектора `ancestor descendant`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-2-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
form {
  border:2px green solid;
  padding:2px;
  margin:0;
  background:#efe;
}
div {
  color:red;
}
fieldset {
  margin:1px;
  padding:3px;
}
</style>
<script type="text/javascript">
$(function(){
  $("form input").css("border", "2px dotted brown");
});
</script>
</head>
<body>
<form>
  <div>Форма заключена в зеленую рамку</div>
  <label>Ребенок:</label>
  <input type="text" name="name" />
  <fieldset>
    <label>Внук:</label>
    <input type="text" name="newsletter" />
  </fieldset>
</form>
Сестринский элемент по отношению к форме: <input type="text" name="none" />
</body>
</html>
```

Обсуждение

В HTML-коде, приведенном в листинге 1.2.1, присутствуют три элемента `input`. Наша задача — отыскать только те из них, которые являются наследниками элемента `form`. Указав в селекторе выражение `form input`, мы легко находим только нужные нам элементы и отмечаем их коричневой рамкой. Элемент `input`, расположенный вне пределов тега `<form>`, такой рамкой отмечен не будет.

Проблема

Необходимо отыскать элементы, являющиеся прямыми потомками какого-либо элемента.

Решение

Используем селектор `parent > child` для решения этой задачи (листинг 1.2.2).

Листинг 1.2.2. Использование селектора `parent > child`

```
<script type="text/javascript">
$(function(){
  $("form > input").css("border", "2px dotted brown");
});
</script>
```

Обсуждение

В листинге 1.2.2 приведен только JavaScript-код, т. к. все остальное осталось без изменений. Поскольку необходимо отыскать только те элементы `input`, которые являются прямыми наследниками `form`, мы решаем задачу, указывая в селекторе выражение `form > input`. Коричневой рамкой в итоге будет отмечен только первый элемент `input`.

Проблема

Необходимо отыскать элементы, следующие непосредственно за известным элементом.

Решение

Используем селектор `prev + next` для решения этой задачи (листинг 1.2.3).

Листинг 1.2.3. Использование селектора `prev + next`

```
<script type="text/javascript">
$(function(){
  $("label + input").css("border", "2px dotted brown");
});
</script>
```

Обсуждение

В листинге 1.2.3 приведен только JavaScript-код. Мы указали в селекторе выражение `prev + next` и, таким образом, нашли элементы `input`, которые следуют непосредственно за элементами `label`.

Проблема

Необходимо отыскать все элементы, располагающиеся на одном уровне (сестринские элементы по отношению друг к другу), следующие непосредственно за известным элементом, который располагается на том же уровне.

Решение

Используем селектор `prev ~ siblings` для решения этой задачи (листинг 1.2.4).

Листинг 1.2.4. Использование селектора `prev ~ siblings`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-2-4</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
form {
  border:2px green solid;
  padding:2px;
  margin:0;
  background:#efe;
}
div {
  color:red;
}
```

```
fieldset {
  margin:1px;
  padding:3px;
}
</style>
<script type="text/javascript">
$(function(){
  $("label ~ fieldset").css("border", "2px dotted brown");
});
</script>
</head>
<body>
<form>
  <div>Форма заключена в зеленую рамку</div>
  <label>Ребенок:</label>
  <input type="text" name="name" />
  <fieldset>
    <label>Внук:</label>
    <input type="text" name="newsletter" />
  </fieldset>
</form>
Сестринский элемент по отношению к форме: <input type="text" name="none"
/>
<fieldset>
  <input type="text" name="email" />
</fieldset>
</body>
</html>
```

Обсуждение

В листинге 1.2.4 мы немного изменили HTML-код, добавив еще один элемент `fieldset`, внутри которого находится элемент `input`. Указываем в селекторе выражение `label ~ fieldset` и видим, что коричневой рамкой отмечен только тот элемент `fieldset`, который находится внутри формы, поскольку только он, в отличие от `fieldset`, находящегося вне `form`, является сестринским элементом по отношению к `label`.

1.3. Основные фильтры

Проблема

Необходимо установить серый цвет фона только для первой строки в таблице.

Решение

Для решения задачи используем селектор `:first` (листинг 1.3.1).

Листинг 1.3.1. Использование фильтра `:first`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-3-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
table {
  width:400px;
}
</style>
<script type="text/javascript">
$(function(){
  $("tr:first").css("background-color","#ccc");
});
</script>
</head>
<body>
<table>
  <tr>
    <td>1-1</td><td>1-2</td><td>1-3</td><td>1-4</td>
  </tr>
  <tr>
    <td>2-1</td><td>2-2</td><td>2-3</td><td>2-4</td>
  </tr>
  <tr>
    <td>3-1</td><td>3-2</td><td>3-3</td><td>3-4</td>
  </tr>
  <tr>
    <td>4-1</td><td>4-2</td><td>4-3</td><td>4-4</td>
  </tr>
</table>
</body>
</html>
```

Обсуждение

HTML-код, приведенный в листинге 1.3.1, не представляет собой ничего интересного — обычная таблица. Посмотрим на JavaScript-код. Указав в селекторе выражение `tr:first` мы смогли установить серый цвет фона только для первой строки таблицы.

Проблема

Необходимо установить серый цвет фона только для последней строки в таблице.

Решение

Для решения задачи используем селектор `:last` (листинг 1.3.2).

Листинг 1.3.2. Использование фильтра `:last`

```
<script type="text/javascript">
$(function(){
  $("tr:last").css("background-color","#ccc");
});
</script>
```

Обсуждение

Рассмотрим только JavaScript-код из листинга 1.3.2, поскольку HTML-код остался без изменений. На этот раз мы указали в селекторе выражение `tr:last` и установили серый цвет фона уже для последней строки в таблице.

Проблема

Необходимо установить серый цвет фона только для четных строк в таблице.

Решение

Для решения задачи воспользуемся селектором `:even` (листинг 1.3.3).

Листинг 1.3.3. Использование фильтра `:even`

```
<script type="text/javascript">
$(function(){
  $("tr:even").css("background-color","#ccc");
});
</script>
```

Обсуждение

Нам стоит только указать выражение `tr:even` в коде, который приведен в листинге 1.3.3, и все четные строки таблицы (отсчет будет идти от нуля) станут серыми.

Проблема

Необходимо установить серый цвет фона только для нечетных строк в таблице.

Решение

Для решения задачи используем селектор `:odd` (листинг 1.3.4).

Листинг 1.3.4. Использование фильтра `:odd`

```
<script type="text/javascript">
$(function(){
  $("tr:odd").css("background-color","#ccc");
});
</script>
```

Обсуждение

Так же просто обстоит дело с отысканием всех нечетных строк. Мы найдем их, всего лишь записав выражение `tr:odd` в коде, который приведен в листинге 1.3.4.

Проблема

Необходимо отыскать шестую по счету ячейку таблицы.

Решение

Для решения задачи воспользуемся селектором `:eq(index)` (листинг 1.3.5).

Листинг 1.3.5. Использование фильтра `:eq(index)`

```
<script type="text/javascript">
$(function(){
  $("td:eq(5)").css("background-color","#ccc");
});
</script>
```

Обсуждение

В примере из листинга 1.3.5 мы нашли шестую ячейку таблицы, указав в селекторе выражение `td:eq(5)`, потому что отсчет идет, начиная от нуля.

Проблема

Необходимо отыскать все ячейки таблицы, которые следуют после шестой по счету (иначе говоря — с индексом, более пяти).

Решение

Для решения задачи используем селектор `:gt(index)` (листинг 1.3.6).

Листинг 1.3.6. Использование фильтра `:gt(index)`

```
<script type="text/javascript">
$(function(){
  $("td:gt(5)").css("background-color","#ccc");
});
</script>
```

Обсуждение

Все очень просто, как код в листинге 1.3.6. Указываем в селекторе выражение `td:gt(5)`, и все ячейки с индексом более пяти стали серого цвета.

Проблема

Необходимо отыскать все ячейки таблицы, которые следуют перед шестой по счету (иначе говоря — с индексом, менее пяти).

Решение

Для решения задачи применим селектор `:lt(index)` (листинг 1.3.7).

Листинг 1.3.7. Использование фильтра `:lt(index)`

```
<script type="text/javascript">
$(function(){
  $("td:lt(5)").css("background-color","#ccc");
});
</script>
```

Обсуждение

Тоже проще простого. Указываем в селекторе выражение `td:lt(5)`, и все ячейки перед шестой стали серыми.

Проблема

Необходимо отыскать на веб-странице все элементы, являющиеся заголовками, например, `h1`, `h2`, `h3` и т. д.

Решение

Для решения задачи используем селектор `:header` (листинг 1.3.8).

Листинг 1.3.8. Использование фильтра `:header`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-3-8</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
table {
  width:400px;
}
</style>
<script type="text/javascript">
$(function(){
  $(":header").css("background-color","#ccc");
});
</script>
</head>
<body>
<h1>Заголовок h1</h1>
<p>Некоторый текст в элементе p</p>
<h3>Заголовок h3</h3>
<ul>
  <li>Список: пункт 1</li>
  <li>Список: пункт 2</li>
  <li>Список: пункт 3</li>
```

```
</ul>
</body>
</html>
```

Обсуждение

В HTML-коде, который приведен в листинге 1.3.8, присутствуют элемент параграф `p`, список `ul` и два элемента, являющиеся заголовками, — `h1` и `h3`. Указывая в селекторе выражение `:header`, мы имеем возможность легко отыскать эти заголовки и установить для них серый цвет фона.

1.4. Фильтрация по содержимому

Проблема

Необходимо отыскать все элементы, внутри которых находится текст, содержащий подстроку 'John'.

Решение

Для решения задачи используем фильтр `:contains` (листинг 1.4.1).

Листинг 1.4.1. Использование фильтра `:contains`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-4-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $("div:contains('John')").css("text-decoration", "underline");
});
</script>
</head>
<body>
<div>John Resig</div>
<div>George Martin</div>
<div>Malcom John Sinclair</div>
<div>J. Ohn</div>
```

```
<div>b byJohns a</div>
</body>
</html>
```

Обсуждение

Рассмотрим пример кода, приведенный в листинге 1.4.1. Имеются пять элементов `div`, внутри которых находится некоторый текст. Чтобы отыскать только те элементы `div`, текст внутри которых содержит подстроку `'John'`, указываем в селекторе выражение `div:contains('John')`, отыскивая только нужные нам элементы. Для наглядности делаем текст внутри этих элементов подчеркнутым с помощью изменения CSS-свойства `text-decoration`.

Проблема

В таблице необходимо отыскать только те ячейки, которые не содержат элементов-потомков, включая и текст.

Решение

Для решения задачи воспользуемся фильтром `:empty` (листинг 1.4.2).

Листинг 1.4.2. Использование фильтра `:empty`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-4-2</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<script type="text/javascript">
$(function(){
  $("td:empty").css("background-color","rgb(255,204,153)")
});
</script>
</head>
<body>
<table style="width:200px;border:1px solid #000;">
  <tr><td>1-1</td><td><span></span></td></tr>
  <tr><td>2-1</td><td></td></tr>
  <tr><td></td><td>3-2</td></tr>
  <tr><td></td><td>4-2</td></tr>
```

```
</table>
</body>
</html>
```

Обсуждение

В примере из листинга 1.4.2 мы применили фильтр `:empty` к элементам `td`, чтобы отыскать пустые ячейки, т. е. ячейки, не имеющие элементов-потомков, в том числе и текстовых узлов. Для найденных ячеек установили цвет фона, используя CSS-свойство `background-color`. Точно так же фильтр `:empty` можно применить и к другим элементам.

Проблема

На веб-странице необходимо отыскать все элементы `div`, внутри которых находится как минимум один элемент `p`.

Решение

Для решения задачи используем фильтр `:has(selector)`(листинг 1.4.3).

Листинг 1.4.3. Использование фильтра `:has(selector)`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-4-3</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
  .test{ border: 3px inset #f00; }
</style>
<script type="text/javascript">
$(function(){
  $("div:has(p)").addClass("test");
});
</script>
</head>
<body>
<div><p>Текст внутри p, который внутри div</p></div>
<div>Это просто текст внутри div</div>
</body>
</html>
```

Обсуждение

В примере из листинга 1.4.3 к элементам `div` мы применили фильтр `:has(p)`. Выбранным оказался элемент `div`, внутри которого находится элемент `p`. Найденный элемент отметили красной рамкой, добавив ему класс с именем `test`.

Проблема

В таблице необходимо отыскать только те ячейки, которые содержат элементы-потомки, включая и текст.

Решение

Для решения задачи воспользуемся фильтром `:parent` (листинг 1.4.4).

Листинг 1.4.4. Использование фильтра `:parent`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-4-4</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
  td { background-color:#393; }
</style>
<script type="text/javascript">
$(function(){
  $("td:parent").fadeTo(3000, 0.3);
});
</script>
</head>
<body>
<table style="width:200px;border:1px solid #000;">
  <tr><td>1-1</td><td><span></span></td></tr>
  <tr><td>2-1</td><td></td></tr>
  <tr><td></td><td>3-2</td></tr>
  <tr><td></td><td>4-2</td></tr>
</table>
</body>
</html>
```

Обсуждение

В примере из листинга 1.4.4 мы применили фильтр `:parent` к элементам `td`, чтобы отыскать непустые ячейки, т. е. ячейки, имеющие элементы-потомки, в том числе и текстовые узлы. К найденным ячейкам таблицы применили метод `fadeTo()`, плавно изменяя значение CSS-свойства `opacity` до 0.3 в течение 3000 миллисекунд. Точно так же фильтр `:parent` можно применить и к другим элементам.

1.5. Фильтры видимых и невидимых элементов

Проблема

На веб-странице необходимо подсчитать число видимых и невидимых элементов и вывести эти значения в окне предупреждения.

Решение

Решим задачу с помощью двух фильтров — `:visible` и `:hidden` (листинг 1.5.1).

Листинг 1.5.1. Использование фильтров `:visible` и `:hidden`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-4-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
div { width:50px; height:50px; margin:5px; border:4px outset #00f;
float:left; }
.starthidden { display:none; }
</style>
<script type="text/javascript">
$(function(){
  alert("Видимых элементов найдено ... " +
  $(":visible", document.body).length +
  "\nНевидимых элементов найдено ... " +
  $(":hidden", document.body).length);
});
```

```
</script>
</head>
<body>
<div></div>
<div class="starthidden"></div>
<div></div>
<div></div>
<div style="display:none;"></div>
<form>
  <input type="hidden" />
  <input type="text" />
  <input type="hidden" />
</form>
</body>
</html>
```

Обсуждение

Код листинга 1.5.1 очень прост. Имеются пять элементов `div`, два из которых скрыты с помощью CSS-свойства `display`, и элемент `form` с тремя элементами `input`, два из которых имеют значение `hidden` в атрибуте `type`, т. е. являются скрытыми полями формы.

В JavaScript-коде мы выводим в окне предупреждения число найденных видимых и невидимых элементов, сопровождая это поясняющим текстом. Интересующие элементы мы ищем в контексте `document.body` из-за того, что фильтр `:hidden` в некоторых браузерах подсчитывает элементы `head`, `title`, `script` и т. д.

В ряде случаев следует учитывать и другие особенности подсчета элементов фильтром `:hidden` — для него считаются невидимыми те элементы (или их родители), которые не имеют размеров в документе. И конечно учитываются элементы, скрытые с помощью CSS.

Итак, в результате выполнения кода из листинга 1.5.1 мы получим сообщение о том, что в теле документа удалось обнаружить пять видимых элементов и четыре невидимых.

1.6. Фильтры атрибутов

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых присутствует атрибут `id`.

Решение

Для решения задачи воспользуемся фильтром `[attribute]` (листинг 1.6.1).

Листинг 1.6.1. Использование фильтра `[attribute]`

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ru" xml:lang="ru">
<head>
<title>example-1-6-1</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="../js/jquery-1.3.2.min.js" type="text/javascript"></script>
<style type="text/css">
div { width:100px; height:100px; margin:5px; padding:5px; border:1px
solid #00f; float:left; }
</style>
<script type="text/javascript">
$(function(){
  $("div[id]").css({ "border-color": "#f00", "color": "#f00" });
});
</script>
</head>
<body>
<div>Элемент div без атрибута id, но с class="example"</div>
<div id="test">Элемент div с атрибутом id="test"</div>
<div id="pretest" class="example">Элемент div с атрибутами id="pretest" и
class="example"</div>
<div>Элемент div без атрибута id</div>
<div id="testend">Элемент div с атрибутом id="testend"</div>
<div id="sometestvalue">Элемент div с атрибутом id="sometestvalue"</div>
<div>Элемент div без атрибута id</div>
<div id="somevalue">Элемент div с атрибутом id="somevalue"</div>
</body>
</html>
```

Обсуждение

В примере из листинга 1.6.1 HTML-код описывает восемь элементов `div`. Одни из них имеют атрибуты `id` или `class`, или даже оба атрибута вместе, а другие не имеют атрибутов вовсе. Этот HTML-код потребуется нам для изучения примеров во всем разделе, посвященном фильтрам атрибутов.

Для того чтобы отыскать все элементы `div`, имеющие атрибут `id`, в селекторе необходимо записать выражение `div[id]`, и задача будет решена. А для наглядности мы установим значения `#f00` для CSS-свойств `border-color` и `color` найденных элементов `div`, окрасив рамку элементов и текст внутри них в красный цвет. "Красными" окажутся второй, третий, пятый, шестой и восьмой элементы `div`.

Проблема

На веб-странице необходимо отыскать элемент `div`, у которого атрибут `id` соответствует определенному значению.

Решение

Для решения задачи воспользуемся фильтром `[attribute=value]` (листинг 1.6.2).

Листинг 1.6.2. Использование фильтра `[attribute=value]`

```
<script type="text/javascript">
$(function(){
  $("div[id=somevalue]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

В примере из листинга 1.6.2 в селекторе мы записываем выражение `div[id=somevalue]`, обнаруживая, таким образом, единственный элемент, у которого атрибут `id` имеет значение `somevalue`. Им оказывается последний, восьмой по счету элемент `div`.

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых атрибут `id` не соответствует определенному значению.

Решение

Для решения задачи применим фильтр `[attribute!=value]` (листинг 1.6.3).

Листинг 1.6.3. Использование фильтра `[attribute!=value]`

```
<script type="text/javascript">
$(function(){
  $("div[id!=somevalue]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

В примере из листинга 1.6.3 мы записали в селекторе выражение `div[id!=somevalue]` и нашли все элементы `div`, у которых значение атрибута `id` отличается от `somevalue`. Обратите внимание, что выбранными оказались также элементы `div`, у которых атрибут `id` и вовсе отсутствует. Таким образом, у нас "покраснели" все элементы `div`, за исключением восьмого.

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых значение атрибута `id` начинается с определенной последовательности символов.

Решение

Для решения задачи воспользуемся фильтром `[attribute^=value]` (листинг 1.6.4).

Листинг 1.6.4. Использование фильтра `[attribute^=value]`

```
<script type="text/javascript">
$(function(){
  $("div[id^=test]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

В примере из листинга 1.6.4, чтобы отыскать все элементы `div`, значение атрибута `id` которых начинается с последовательности символов `test`, мы запи-

сали в селекторе выражение `div[id^=somevalue]`. "Красными" стали второй и пятый элементы `div`.

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых значение атрибута `id` заканчивается определенной последовательностью символов.

Решение

Для решения задачи воспользуемся фильтром `[attribute$=value]` (листинг 1.6.5).

Листинг 1.6.5. Использование фильтра `[attribute$=value]`

```
<script type="text/javascript">
$(function(){
  $("div[id$=test]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

В примере из листинга 1.6.5, чтобы отыскать все элементы `div`, значение атрибута `id` которых заканчивается последовательностью символов `test`, мы записали в селекторе выражение `div[id$=somevalue]`. "Красными" стали второй и третий элементы `div`.

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых значение атрибута `id` содержит определенную последовательность символов.

Решение

Для решения задачи воспользуемся фильтром `[attribute*=value]` (листинг 1.6.6).

Листинг 1.6.6. Использование фильтра `[attribute*=value]`

```
<script type="text/javascript">
$(function(){
  $("div[id*=test]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

В примере из листинга 1.6.6, чтобы отыскать все элементы `div`, значение атрибута `id` которых содержит последовательность символов `test`, мы записали в селекторе выражение `div[id*=somevalue]`. "Красными" стали второй, третий, пятый и шестой элементы `div`.

Проблема

На веб-странице необходимо отыскать элементы `div`, у которых значение атрибута `id` содержит определенную последовательность символов и к тому же присутствует атрибут `class` с заданным значением.

Решение

Решим задачу с помощью фильтра `[attributeFilter1][attributeFilter2]...[attributeFilterN]` (листинг 1.6.7).

Листинг 1.6.7. Использование фильтра `[attributeFilter1][attributeFilter2]...[attributeFilterN]`

```
<script type="text/javascript">
$(function(){
  $("div[id*=test][class=example]").css({
    "border-color": "#f00", "color": "#f00"
  });
});
</script>
```

Обсуждение

Чтобы отыскать все элементы `div`, значение атрибута `id` которых содержит последовательность символов `test` и одновременно атрибут `class` которых