Лабораторная работа №6
Основы jQuery

Цель: научиться использовать библиотеку jQuery.

jQuery — библиотека, которая позволяет делать код короче, а также позволяет внутри страницы настроить код, который бы **!!! срабатывал как триггер (предопределенный набор действий, который выполняется автоматически при наступлении связанного с ним события, если этот код описываем в области <head> … </head>).**

JQuery библиотека содержит следующий функционал:

- операции с HTML/DOM (манипулирование компонентами HTML/DOM)
- операции с CSS-селекторами
- HTML-обработчики событий
- Эффекты анимации
- AJAX
- Utilities

JQuery упрощает работу с JavaScript, а также вызовы AJAX и DOM-манипуляции.
Есть много фреймворков JavaScript, но JQuery, является самым популярным и используемым за счет своей расширяемости.

Начало работы с библиотекой

Библиотеку jQuery можно скачать с сайта http://jquery.com, а можно вставить в документ, используя известные интернет-адреса:

- По адресу `http://code.jquery.com/jquery-latest.js` – доступна всегда последняя версия.
- С Google:
  https://developers.google.com/speed/libraries/devguide?hl=ru#jquery можно загрузить любую из не слишком старых версий. Синтаксис такой: `src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"`, где `1.8.3` — версия, причём можно указать её приблизительно: `1.8` означает последнюю версию вида `1.8.*`, а `1` — последнюю версию вида `1.*`. Файл `jquery.min.js` обозначает сжатый код, а `jquery.js` — несжатый, для удобства отладки.
- Либо с Microsoft CDN:
  `src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.9.1.min.js"`

Пример подключения ниже (!!! но использовать один из вариантов подключения библиотеки):

```
<!DOCTYPE html>
<html>
<head>
<!--если выбираем библиотеку от Microsoft CDN -->
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.9.1.min.js"></script>
<!--если выбираем библиотеку от Google -->
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>
<!--третий вариант -->
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
</body>
</html>
```

jQuery синтаксис

Базовая команда для библиотеки вяглядит как: **$(*селектор*).*action*()**, где

- $ - предписание использовать jQuery;
- (*селектор*) - это "запрос или элементы поиска" в HTML элементах страницы;
- *action*() - это действия, которые должны быть выполнены над найденными элементами (это те элементы, которые удовлетворяют условиям селектора).

Например:
$(this).hide() – скрывает текущий элемент (где **this** – это указатель на текущий элемент, позволяет делать код универсальным за счет того, что не надо писать здесь имя или id элемента, над которым будет производится действие hide()).

$("p").hide() – скрывает все <p> элементы на странице.

$(".test").hide() – скрывает все элементы на странице, которые ассоциированы с классом "test".

$("#test").hide() – скрывает все элементы на странице, у которых id="test".

Событие Ready у объекта страницы Document

Вы, заметите, что в большинстве примеров jQuery-методы находятся внутри события документа Ready():

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

  // jQuery-методы размещаем здесь...
});
```

Это необходимо для предотвращения любых срабатываний JQuery-кода, прежде чем документ не закончит полную загрузку.
Это хорошая практика, чтобы дождаться, пока документ будет полностью загружен и готов до работы с ним. Это также позволяет вам сформировать свой JavaScript код в головной части, прежде чем тело документа.

jQuery селекторы

jQuery селекторы позволяют вам делать выборку (поиск) и манипулировать с элементами HTML. Селекторы по сути это набор условных обозначений и правил для выборки и манипулирования (в конце лабораторной в приложении дан большой список примеров селекторов).
С jQuery селекторами вы можете найти элементы страницы, основанные на идентификаторе id, классах (class), типах (type), атрибутах (attribute), значениях атрибутов (value) и др. Также они базируются и на CSS Selectors, в дополнении вы можете создать свой селектор. Все типы селекторов в jQuery начинаются с указания $ и парных скобок:

$( )

Например, в следующем коде при нажатии на кнопку выполняется поиск на странице всех элементов, обозначенных тегом <p> , и все эти элементы скрываются на странице (срабатывает метод hide( )):

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
```

```
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").hide();
        });
    });
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Обратите внимание на то, что селектор в $( )указывается в кавычках а не в < >.

Задание 1: Выполнить пример, проверить работу метода hile ( ). Попробовать заменить библиотеку `jquery-latest.js` на другие две, описанные выше, сравнить результаты работы.

Пояснения к скрипту:

```
<script>

//в строке ниже в качестве селектора использован весь объект –документ,
//при этом будет срабатывать jquery-запрос, как только наступит событие полной готовности
//страницы к работе с пользователем и запустится метод ready(),
// при срабатывании которого будет создана следующая функция

    $(document).ready(function () {

//созданная функция в свою очередь будет jquery-запросом, который ищет все элементы типа
//button – кнопка, и с их методами click связывает (ассоциирует) действие в виде функции,

        $("button").click(function () {

// которая выполнит jquery-запрос, который для всех найденных элементов внутри тега <p>
выполнит метод hide(), т.е. скроет их со страницы

        $("p").hide();

//далее закрываем внутреннюю функцию

        });

//далее закрываем внешнюю функцию

    });
</script>
```

В таком исполнении скрипт, помещенный в заголовок страницы внутрь метода объект document.ready( ) работает как триггер, т.е. автоматически срабатывает при наступлении определенного события на странице. В примере – это событие click( ) кнопки. А так как в методе hide( ) нет никакого описания другого кода, то он выполняет те действия, для которых он изначально создан, а именно скрывает объект.

Селектор #id

Селектор jQuery #id использует id атрибут в HTML-тегах, чтобы найти определенный элемент.

Id должен быть уникальным внутри всей страницы, если вы хотите найти с его помощью конкретный уникальный элемент.

Чтобы найти элемент с помощью id, то перед названием искомого идентификатора ставится знак #, например:

$("#test")

В примере ниже, при нажатии на кнопку ищется элемент с идентификатором test и скрывается со страницы.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("#test").hide();
        });
    });
</script>
</head>

<body>
<h2>This is a heading 2</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button>Click me</button>
</body>

</html>
```

Задание 2: Выполнить пример, проанализировать отличие его от примера в задании 1.

Ниже приведен пример, в котором работают две кнопки: одна – скрывает элемент с id=”test”, вторая – отображает этот элемент (используется метод show( )). Обратите внимание, чтобы распараллелить код по двум кнопкам, для каждой из них тоже были определены id, по которым определяется какую функцию запускать.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("#but1").click(function () {
            $("#test").hide();
        });
        $("#but2").click(function () {
            $("#test").show();
        });
    });
</script>
</head>

<body>
<h2>This is a heading 2</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button id="but1">Click me</button>
<button id="but2">Click me for show</button>
</body>

</html>
```

Задание3: Выполнить пример, проанализировать отличие его от примера в задании 2.

.class селектор

Селектор jQuery class находит элементы определенного класса.

Для поиска элементов определенного класса указывается перед названием точка, например:

$(".test")

В примере ниже при нажатии на кнопку элементы с классом="test" будут скрыты:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $(".test").hide();
        });
    });
</script>
</head>
<body>

<h2 class="test">This is a heading</h2>
<p class="test">This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Задание 4: Выполнить пример, проанализировать отличие его от примера в задании 3.

Другие примеры jQuery селекторов (делать необязательно, просто при необходимости для ознакомления)

Синтаксис	Описание	Пример
$("*")	Selects all elements	Try it
$(this)	Selects the current HTML element	Try it
$("p.intro")	Selects all <p> elements with class="intro"	Try it
$("p:first")	Selects the first <p> element	Try it
$("ul li:first")	Selects the first element of the first	Try it
$("ul li:first-child")	Selects the first element of every	Try it
$("[href]")	Selects all elements with an href attribute	Try it
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"	Try it
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"	Try it
$(":button")	Selects all <button> elements and <input> elements of type="button"	Try it
$("tr:even")	Selects all even <tr> elements	Try it
$("tr:odd")	Selects all odd <tr> elements	Try it

Индивидуальное задание 1:

Вариант 1: в исходный файл Experiments.html добавить в начало две кнопки: одну-для скрытия элементов, другую – для отображения скрытых элементов. Настроить методы click( ) кнопок, так чтобы они то скрывали, то отображали нечетные элементы с классом `MsoNormal.`

Вариант 2: в исходный файл Experiments.html добавить в начало две кнопки: одну-для скрытия элементов, другую – для отображения скрытых элементов. Настроить методы click( ) кнопок, так чтобы они то скрывали, то отображали четные элементы типа `<tr>.`

Вариант 3: в исходный файл Experiments.html добавить в начало две кнопки: одну-для скрытия элементов, другую – для отображения скрытых элементов. Настроить методы click( ) кнопок, так чтобы они то скрывали, то отображали элементы с атрибутом `href.`

Вариант 4: в исходный файл Experiments.html добавить в начало две кнопки: одну-для скрытия элементов, другую – для отображения скрытых элементов. Настроить методы click( ) кнопок, так чтобы они то скрывали, то отображали элементы, у которых атрибут `align` равен значению `center.`

jQuery-методы событий

Ниже представлены общие с технологией DOM события:

События мыши	**События клавиатуры**	**События формы**	**События документа/окна**
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

jQuery-синтаксис для методой событий

В jQuery, большинство DOM-событий имеют эквивалентный jQuery-метод.
Чтобы назначить событие нажатия мышкой на все элементы <p> на странице, вы можете написать:

```
$("p").click();
```

Следующий шаг – это определение того, что будет происходить, когда наступит указанное событие.
Вы должны определить функцию для события:

```
$("p").click(function(){
  // описание действий в функции
});
```

Часто используемые методы JQuery

$(document).ready( )

$(document).ready() метод позволяет вам выполнить функцию, когда документ полностью загружен.

click( )

Эта функция выполняется, когда пользователь нажимает на HTML элемент.

В примере событие click( ) активируется на <p> элементах, скрывая текущий <p> элемент:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("p").click(function () {
```

```
            $(this).hide();
        });
    });
</script>
</head>
<body>

<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>
</body>
</html>
```

Задание 5: Выполнить пример, проверить работу.

Обратите внимание! `$(this).hide();//указатель this позволяет выполнять действие //с текущим активированным элементом`

dblclick( )

Срабатывает, когда пользователь двойным щелчком нажимает на HTML-элемент:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("p").dblclick(function () {
            $(this).hide();
        });
    });
</script>
</head>
<body>

<p>If you double-click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

Задание 6: Выполнить пример, проверить работу.

mouseenter()

Выполняется, когда указатель мыши наводится на HTML-элемент:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("#p1").mouseenter(function () {
            alert("You entered p1!");
        });
    });
</script>
</head>
<body>

<p id="p1">Enter this paragraph.</p>
```

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

Задание 7: Выполнить пример, проверить работу.

blur()

Выполняется, когда поле формы теряет фокус.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("input").focus(function () {
            $(this).css("background-color", "#cccccc");
        });
        $("input").blur(function () {
            $(this).css("background-color", "#ffffff");
        });
    });
</script>
</head>
<body>

Name: <input type="text" name="fullname"><br>
Email: <input type="text" name="email">

</body>
</html>
```

Задание 8: Выполнить пример, проверить работу. Проверить как работает focus() и blur(), а также разобраться с настройкой стилей через .css("свойство", "значение").

hide() и show()

С jQuery вы можете скрывать и отражать HTML-элементы:
Также возможно настроить время затухания и появления:
Синтаксис:
$(*selector*).hide(*speed*);

$(*selector*).show(*speed*);
или
$(*selector*).hide(*speed,callback*);

$(*selector*).show(*speed,callback*);

Необязательный параметр скорости определяет скорость скрытия / показа, и может принимать следующие значения: "slow", "fast" или в миллисекундах.

Следующий пример демонстрирует параметр скорости скрытия элемента:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").hide(1000);
        });
    });
```

```
</script>
</head>
<body>
<button>Hide</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

Задание 9: Выполнить пример, проверить работу.

toggle()

С помощью метода можно переключать выполнение методов hide() и show(). Также можно настроить и скорость срабатывания.

Синтаксис:

$(*selector*).toggle(*speed*);

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").toggle();
        });
    });
</script>
</head>
<body>

<button>Toggle</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

Задание 10: Выполнить пример, проверить работу.

Индивидуальное задание 2:

Вариант 1: в исходный файл Experiments2.html добавить обработку события «наведение мышки на элемент». Если этот элемент относится к таблице (в теге <img>), то при наведении на него курсора, он должен скрываться.

Вариант 2: в исходный файл Experiments2.html добавить обработку события «двойной щелчок мышки по элементу». Для элемента с атрибутом `align="center"` нужно настроить исчезновение/появление элемента.

Вариант 3: в исходный файл Experiments2.html добавить обработку события «изменение размера окна». При срабатывании этого метода скрыть с документа все элемента со ссылками (href).

Вариант 4: в исходный файл Experiments2.html добавить элемент формы «поле ввода», настроить для него обработку keypress, а именно случайным образом менять цвет этого поля (свойство `background-color`).

Fading методы

С JQuery вы можете исключать и включать видимость элементов.

Набор методов:

- fadeIn()

- fadeOut()
- fadeToggle()
- fadeTo()

fadeIn()

fadeIn() используется, чтобы реализовать проявление в скрытых элементах (например, элемент скрытый).

Можно также использовать настройку скорости срабатывания.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("#div1").fadeIn();
            $("#div2").fadeIn("slow");
            $("#div3").fadeIn(3000);
        });
    });
</script>
</head>

<body>
<p>Demonstrate fadeIn() with different parameters.</p>
<button>Click to fade in boxes</button>
<br><br>
<div id="div1" style="width:80px;height:80px;display:none;background-
color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;display:none;background-
color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;display:none;background-color:blue;"></div>

</body>
</html>
```

Задание 11: Выполнить пример, проверить работу. Разобраться с настройками элементов , обратите внимание на свойство `display:none` , которое при загрузке странице позволяет не отражать .

fadeOut()

fadeOut() используется, чтобы реализовать скрытие элементов из зоны видимости.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("#div1").fadeOut();
            $("#div2").fadeOut("slow");
            $("#div3").fadeOut(3000);
        });
    });
</script>
</head>

<body>
<p>Demonstrate fadeOut() with different parameters.</p>
```

```
<button>Click to fade out boxes</button>
<br><br>
<div id="div1" style="width:80px;height:80px;background-color:red;"></div><br>
<div id="div2" style="width:80px;height:80px;background-color:green;"></div><br>
<div id="div3" style="width:80px;height:80px;background-color:blue;"></div>

</body>
</html>
```

Задание 12: Выполнить пример, проверить работу. Обратите внимание на отсутствие свойства `display:none` , поэтому настройки элементов сначала отражаются.

Индивидуальное задание 3:

Вариант 1: создать файл Individual_3.html, добавить два рисунка, невидимые при загрузке страницы, и две кнопки. При нажатии на одну кнопку должно в цикле срабатывать проявление первой картинки и исчезновение второй картинки со скоростью 2000, а затем наоборот, при нажатии на вторую кнопку обе картинки должны появиться и прекратить мигание.

Вариант 2: создать файл Individual_3.html, добавить пять областей div с id “div1”, “div2”,…,“div5”, в которые поместить тексты, невидимые при загрузке страницы, и две кнопки. При нажатии на одну кнопку должно определяться случайное число от 1 до 5. В зависимости от этого числа должна проявиться соответствующая область div со скоростью 3000. При нажатии на вторую кнопку все области div снова скрываются.

Вариант 3: создать файл Individual_3.html, добавить 10 параграфов (<p>) с каким-либо текстом, невидимые при загрузке страницы, и кнопку. При нажатии на кнопку должно в цикле срабатывать проявление четных параграфов и исчезновение нечетных со скоростью 2000, а затем наоборот.

Вариант 4: создать файл Individual_3.html, добавить пять областей div 40x40 пикселей с id “div1”, “div2”,…,“div5”, раскрашенные в разные цвета, невидимые при загрузке страницы, и кнопку. При нажатии на кнопку должно в цикле срабатывать волнообразное проявление областей div слево-направо, а затем наоборот, исчезновение. Цикл работает до тех пор, пока не закрыта страница.

Sliding методы

Реализацию движения по принципу слайдов можно выполнять с помощью методов:

- slideDown()
- slideUp()
- slideToggle()

slideDown()

Используется для скольжения элемента вниз.

Синтаксис:

$(*selector*).slideDown(*speed*);

Необязательный параметр скорости определяет скорость скольжения, и может принимать следующие значения: "slow", "fast" или в миллисекундах.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("#flip").click(function () {
            $("#panel").slideDown("slow");
        });
    });
```

```
</script>

<style type="text/css">
#panel,#flip
{
padding:5px;
text-align:center;
background-color:#e5eecc;
border:solid 1px #c3c3c3;
}
#panel
{
padding:50px;
display:none;
}
</style>
</head>
<body>

<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```

Задание 13: Выполнить пример, проверить работу. Разобраться с элементом `<style type="text/css">`. Обратите внимание, что настройки делаются как общие для нескольких элментов, так и индивидуальные.

Метод animate()

Animate() используется для создания пользовательской анимации.

Синтаксис:

$(*selector*).animate({*params*},*speed*,*callback*);

Обязательный ***params*** определяется параметрами CSS-свойств для анимации.

Необязательный параметр ***speed*** определяет длительность эффекта. Он может принимать следующие значения: "slow", "fast", или в миллисекундах.

В следующем примере показан простой вариант использования метода, он перемещает элемент влево, пока он не достиг значения свойства **left**=250px:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("div").animate({ left: '250px' });
        });
    });
</script>
</head>
<body>
<button>Start Animation</button>
<p>By default, all HTML elements have a static position, and cannot be moved. To
manipulate the position, remember to first set the CSS position property of the element
to relative, fixed, or absolute!</p>
<div style="background:#98bf21;height:100px;width:100px;position:absolute;">
</div>
</body>
</html>
```

Задание 14: Выполнить пример, проверить работу. Разобраться с элементом `<div>`.

С прочими примерами анимации можно познакомиться на
http://www.w3schools.com/jquery/jquery_animate.asp

Перебор результатов поиска

Результатом поиска является jQuery-объект. Он похож на массив: в нём есть нумерованные элементы и length (длина массива), но методы у него совсем другие.
jQuery-объект также называют «jQuery-коллекцией», «элементами, обёрнутыми в jQuery» и пр. Используем jQuery, чтобы выбрать все элементы по селектору `li> a` и перебрать их:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery.js"></script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            var links = $('li> a');
            // перебор результатов
            for (var i = 0; i < links.length; i++) {
                alert(links[i].href);
            }

        });
    });
</script>
</head>

<body>
<ul>
  <li><a href="http://jquery.com">jQuery</a></li>
  <li><a href="http://jqueryui.com">jQuery UI</a></li>
  <li><a href="http://blog.jquery.com">jQuery Blog</a></li>
</ul>
<button>Start query</button>

</body>
</html>
```

Задание 15: Выполнить пример, проверить работу. Разобраться, как организована работа с переменной **links** и работа с ней в цикле.

Контекст поиска

А что, если контекст поиска содержит много элементов? Например, как будет работать запрос `$('a', 'li')`, если `<li>` в документе много?
Если в контексте много элементов, то поиск будет произведён в каждом из них. А затем результаты будут объединены.
Повторы элементов при этом отфильтровываются, то есть два раза один и тот же элемент в результате не появится.
Например, найдём `$('a', 'li')` в многоуровневом списке:

```
<!DOCTYPE HTML>
<html>
<body>
<script src="http://code.jquery.com/jquery.js"></script>

<ul>
  <li>
    <a href="http://jquery.com">jQuery</a>
    <ul>
      <li><a href="http://blog.jquery.com">jQuery Blog</a></li>
    </ul>
  </li>
  <li><a href="http://sizzlejs.com">Sizzle</a></li>
```

```
</ul>

<script>

    var links = $('a', 'li');
    for (var i = 0; i < links.length; i++) {
        alert(i + ": " + links[i].href); // 3 ссылки по очереди
    }
</script>

</body>
</html>
```

Задание 16: Выполнить пример, проверить работу. Разобраться, как организована работа jQuery, когда скрипт размещается в теле страницы (нет объявления функций и привязок к методам).

!!!Обратите внимание, что здесь jQuery-запрос идет в теле документа без привязки к событию, поэтому работает в режиме немедленного запуска (а не срабатывания по принципу триггера на событие в загруженном документе).

Метод each()

Для более удобного перебора у jQuery-коллекции есть метод each. Его синтаксис похож на forEach массива:

```
.each( function(index, item) )
```

Он выполняет для каждого элемента коллекции перед точкой функцию-аргумент, и передаёт ей номер `index` и очередной элемент `item`.

Используем его вместо `for`, чтобы перебрать коллекцию найденных ссылок:

```
$('li a').each(function(i, a) {
  alert( i + ": " + a.href);
});
```

У **`.each`** есть важная возможность, которой нет в **`forEach`**: возврат **`false`** из функции прекращает перебор.

Например:

```
<!DOCTYPE HTML>
<html>
<body>
<script src="http://code.jquery.com/jquery.js"></script>

<a href="http://wikipedia.ru">Википедия</a>

<ul>
  <li><a href="http://jquery.com">jQuery</a></li>
  <li><a href="http://sizzlejs.com">Sizzle</a></li>
  <li><a href="http://blog.jquery.com">jQuery Blog</li>
</ul>
<script>
    var links = $('li a'); // найти все ссылки на странице внутри LI

    links.each(function (i, a) {
        alert(i + ': ' + a.innerHTML);

        if (i == 1) return false; // стоп на элементе коллекции с индексом 1
    });
</script>
</body>
</html>
```

Задание 17: Выполнить пример, проверить работу. Разобраться, как организована работа метода each( ).

Получение контента с помощью text(), html(), and val()

Три основных, но полных jQuery метода для манипулирования данными технологии DOM:

- text() - Задает или возвращает текстовое содержимое выбранных элементов.
- html() - Задает или возвращает содержание отдельных элементов (включая HTML markup).
- val() - Задает или возвращает значение поля формы.

В следующем примере показано, как получить контент с помощью text() и html() методов:

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery.js"></script>
<script>
    $(document).ready(function () {
        $("#btn1").click(function () {
            alert("Text: " + $("#test").text());
        });
        $("#btn2").click(function () {
            alert("HTML: " + $("#test").html());
        });
    });
</script>
</head>

<body>
<p id="test">This is some <b>bold</b> text in a paragraph.</p>
<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>
</body>
</html>
```

Задание 18: Выполнить пример, проверить работу. Разобраться, как организована работа метода `text()` и `html()`.

В следующем примере показано, как получить значение поля ввода с помощью метода VAL():

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery.js"></script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            alert("Value: " + $("#test").val());
        });
    });
</script>
</head>

<body>
<p>Name: <input type="text" id="test" value="Mickey Mouse"></p>
<button>Show Value</button>
</body>
</html>
```

Задание 19: Выполнить пример, проверить работу. Разобраться, как организована работа метода `val()`.

Получение значений атрибутов с помощью attr()

В следующем примере показано, как получить значение атрибута HREF из ссылки:

```
<!DOCTYPE html>
```

```
<html>
<head>
<script src="http://code.jquery.com/jquery.js"></script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            alert($("#w3s").attr("href"));
        });
    });
</script>
</head>

<body>
<p><a href="http://www.w3schools.com" id="w3s">W3Schools.com</a></p>
<button>Show href Value</button>
</body>
</html>
```

Задание 20: Выполнить пример, проверить работу. Разобраться, как организована работа метода `attr()`. Определите, значения каких еще атрибутов можно так получить.

Индивидуальное задание 4:

Вариант 1: создать файл Individual_4.html, добавить сверху кнопку, а затем пять параграфов с каким-либо текстом и между каждым параграфом по маленькому рисунку. При нажатии на кнопку среди всех найденных параграфов для 3-го и 5-го изменить положение – сместить его на 100 пт влево относительно текущего положения. При нажатии на любую из картинок должно происходить движение текущей вниз на 30 пт.

Вариант 2: создать файл Individual_4.html, добавить сверху кнопку, а затем 6 маленьких рисунков одинакового размера, один под другим. При нажатии на кнопку среди всех найденных рисунков для четных выполнить эффект слайд-шоу – заезд/скрытие рисунка под соответствующим ему верхним рисунком со скоростью «slow».

Вариант 3: создать файл Individual_4.html, добавить сверху кнопку, а затем 6 параграфов с каким-либо текстом, один под другим. При нажатии на кнопку среди всех найденных рисунков для нечетных выполнить извлечение текста и присвоение этого текста соответствующему нижнему/четному параграфу и сместить нечетные элементы на 40 пт влево.

Вариант 4: создать файл Individual_4.html, добавить сверху кнопку, а затем 6 маленьких рисунков одинакового размера, один под другим. При нажатии на кнопку среди всех найденных рисунков для нечетных выполнить эффект слайд-шоу – заезд/скрытие рисунка под соответствующим ему нижним рисунком со скоростью «slow».

Получение конкретного элемента

Даже если элемент найден только один, всё равно результатом будет jQuery-коллекция.
Для получения одного элемента из jQuery-коллекции есть несколько способов:

1. Метод get(индекс), работает так же, как прямой доступ:

```
alert( $('body').get(0) ); // BODY
```

Если элемента с таким номером нет — вызов `get` возвратит `undefined`.

2. Метод eq(индекс) возвращает коллекцию из одного элемента — с данным номером. Он отличается от метода get(индекс) и прямого обращения по индексу тем, что возвращает именно jQuery-коллекцию с одним элементом, а не сам элемент.

```
// DOM-элемент для первой ссылки
$('a').get(0);
// jQuery-объект из одного элемента: первой ссылки
$('a').eq(0);
```

Во втором случае вызов `eq` создаёт новую jQuery-коллекцию, добавляет в нее нулевой элемент и возвращает. Это удобно, если мы хотим дальше работать с этим элементом, используя методы jQuery. Если элемента с таким номером нет — `eq` возвратит пустую коллекцию.

Отчет по лабораторной работе

В соответствии со структурой заготовки отчета и примером оформления оформить в отчете все задания, выполняемые в ходе лабораторной работы, а также индивидуальные задания по вариантам. Файл с отчетом называть по шаблону: **Фамилия_лаб_раб_номер.**

Отчет предоставляется в электронном виде либо лично преподавателю, либо на электронную почту для проверки. Также по результатам лабораторной работы на следующем за ней занятии проводится выборочный опрос по командам языка.

jQuery Selectors

Use our excellent jQuery Selector Tester to experiment with the different selectors.

Selector	Example	Selects
*	$("*")	All elements
#id	$("#lastname")	The element with id="lastname"
.class	$(".intro")	All elements with class="intro"
.class,.class	$(".intro,.demo")	All elements with the class "intro" or "demo"
element	$("p")	All <p> elements
el1,el2,el3	$("h1,div,p")	All <h1>, and <p> elements
:first	$("p:first")	The first <p> element
:last	$("p:last")	The last <p> element
:even	$("tr:even")	All even <tr> elements
:odd	$("tr:odd")	All odd <tr> elements
:first-child	$("p:first-child")	All <p> elements that are the first child of their parent
:first-of-type	$("p:first-of-type")	All <p> elements that are the first <p> element of their parent
:last-child	$("p:last-child")	All <p> elements that are the last child of their parent
:last-of-type	$("p:last-of-type")	All <p> elements that are the last <p> element of their parent
:nth-child(n)	$("p:nth-child(2)")	All <p> elements that are the 2nd child of their parent
:nth-last-child(n)	$("p:nth-last-child(2)")	All <p> elements that are the 2nd child of their parent, counting from the last child
:nth-of-type(n)	$("p:nth-of-type(2)")	All <p> elements that are the 2nd <p> element of their parent
:nth-last-of-type(n)	$("p:nth-last-of-type(2)")	All <p> elements that are the 2nd <p> element of their parent, counting from the last child
:only-child	$("p:only-child")	All <p> elements that are the only child of their parent
:only-of-type	$("p:only-of-type")	All <p> elements that are the only child, of its type, of their parent
parent > child	$("div > p")	All <p> elements that are a direct child of a element
parent descendant	$("div p")	All <p> elements that are descendants of a element
element + next	$("div + p")	The <p> element that are next to each elements
element ~ siblings	$("div ~ p")	All <p> elements that are siblings of a element
:eq(index)	$("ul li:eq(3)")	The fourth element in a list (index starts at 0)
:gt(no)	$("ul li:gt(3)")	List elements with an index greater than 3
:lt(no)	$("ul li:lt(3)")	List elements with an index less than 3
:not(selector)	$("input:not(:empty)")	All input elements that are not empty
:header	$(":header")	All header elements <h1>, <h2> ...
:animated	$(":animated")	All animated elements
:focus	$(":focus")	The element that currently has focus
:contains(text)	$(":contains('Hello')")	All elements which contains the text "Hello"
:has(selector)	$("div:has(p)")	All elements that have a <p> element
:empty	$(":empty")	All elements that are empty
:parent	$(":parent")	All elements that are a parent of another element
:hidden	$("p:hidden")	All hidden <p> elements
:visible	$("table:visible")	All visible tables
:root	$(":root")	The document's root element
:lang(language)	$("p:lang(de)")	All <p> elements with a lang attribute value starting with "de"
[attribute]	$("[href]")	All elements with a href attribute
[attribute=value]	$("[href='default.htm']")	All elements with a href attribute value equal to "default.htm"
[attribute!=value]	$("[href!='default.htm']")	All elements with a href attribute value not equal to "default.htm"
[attribute$=value]	$("[href$='.jpg']")	All elements with a href attribute value ending with ".jpg"

[attribute|=value]	$("[hreflang|='en']")	All elements with a hreflang attribute value starting with "en"
[attribute^=value]	$("[name^='hello']")	All elements with a name attribute value starting with "hello"
[attribute~=value]	$("[name~='hello']")	All elements with a name attribute value containing the word "hello"
[attribute*=value]	$("[name*='hello']")	All elements with a name attribute value containing the string "hello"
:input	$(":input")	All input elements
:text	$(":text")	All input elements with type="text"
:password	$(":password")	All input elements with type="password"
:radio	$(":radio")	All input elements with type="radio"
:checkbox	$(":checkbox")	All input elements with type="checkbox"
:submit	$(":submit")	All input elements with type="submit"
:reset	$(":reset")	All input elements with type="reset"
:button	$(":button")	All input elements with type="button"
:image	$(":image")	All input elements with type="image"
:file	$(":file")	All input elements with type="file"
:enabled	$(":enabled")	All enabled input elements
:disabled	$(":disabled")	All disabled input elements
:selected	$(":selected")	All selected input elements
:checked	$(":checked")	All checked input elements

jQuery Event Methods

Event methods trigger or attach a function to an event handler for the selected elements.
The following table lists all the jQuery methods used to handle events.

Method	Description
bind()	Attaches event handlers to elements
blur()	Attaches/Triggers the blur event
change()	Attaches/Triggers the change event
click()	Attaches/Triggers the click event
dblclick()	Attaches/Triggers the double click event
delegate()	Attaches a handler to current, or future, specified child elements of the matching elements
die()	Removed in version 1.9. Removes all event handlers added with the live() method
error()	Attaches/Triggers the error event
event.currentTarget	The current DOM element within the event bubbling phase
event.data	Contains the optional data passed to an event method when the current executing handler is bound
event.delegateTarget	Returns the element where the currently-called jQuery event handler was attached
event.isDefaultPrevented()	Returns whether event.preventDefault() was called for the event object
event.isImmediatePropagationStopped()	Returns whether event.stopImmediatePropagation() was called for the event object
event.isPropagationStopped()	Returns whether event.stopPropagation() was called for the event object
event.namespace	Returns the namespace specified when the event was triggered
event.pageX	Returns the mouse position relative to the left edge of the document
event.pageY	Returns the mouse position relative to the top edge of the document
event.preventDefault()	Prevents the default action of the event
event.relatedTarget	Returns which element being entered or exited on mouse movement.
event.result	Contains the last/previous value returned by an event handler triggered by the specified event
event.stopImmediatePropagation()	Prevents other event handlers from being called
event.stopPropagation()	Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event
event.target	Returns which DOM element triggered the event
event.timeStamp	Returns the number of milliseconds since January 1, 1970, when the event is triggered

event.type	Returns which event type was triggered
event.which	Returns which keyboard key or mouse button was pressed for the event
focus()	Attaches/Triggers the focus event
focusin()	Attaches an event handler to the focusin event
focusout()	Attaches an event handler to the focusout event
hover()	Attaches two event handlers to the hover event
keydown()	Attaches/Triggers the keydown event
keypress()	Attaches/Triggers the keypress event
keyup()	Attaches/Triggers the keyup event
live()	Removed in version 1.9. Adds one or more event handlers to current, or future, selected elements
load()	Attaches an event handler to the load event
mousedown()	Attaches/Triggers the mousedown event
mouseenter()	Attaches/Triggers the mouseenter event
mouseleave()	Attaches/Triggers the mouseleave event
mousemove()	Attaches/Triggers the mousemove event
mouseout()	Attaches/Triggers the mouseout event
mouseover()	Attaches/Triggers the mouseover event
mouseup()	Attaches/Triggers the mouseup event
off()	Removes event handlers attached with the on() method
on()	Attaches event handlers to elements
one()	Adds one or more event handlers to selected elements. This handler can only be triggered once per element
$.proxy()	Takes an existing function and returns a new one with a particular context
ready()	Specifies a function to execute when the DOM is fully loaded
resize()	Attaches/Triggers the resize event
scroll()	Attaches/Triggers the scroll event
select()	Attaches/Triggers the select event
submit()	Attaches/Triggers the submit event
toggle()	Removed in version 1.9. Attaches two or more functions to toggle between for the click event
trigger()	Triggers all events bound to the selected elements
triggerHandler()	Triggers all functions bound to a specified event for the selected elements
unbind()	Removes an added event handler from selected elements
undelegate()	Removes an event handler to selected elements, now or in the future
unload()	Attaches an event handler to the unload event

jQuery Effect Methods

The following table lists all the jQuery methods for creating animation effects.

Method	Description
animate()	Runs a custom animation on the selected elements
clearQueue()	Removes all remaining queued functions from the selected elements
delay()	Sets a delay for all queued functions on the selected elements
dequeue()	Removes the next function from the queue, and then executes the function
fadeIn()	Fades in the selected elements
fadeOut()	Fades out the selected elements
fadeTo()	Fades in/out the selected elements to a given opacity
fadeToggle()	Toggles between the fadeIn() and fadeOut() methods
finish()	Stops, removes and completes all queued animations for the selected elements
hide()	Hides the selected elements
queue()	Shows the queued functions on the selected elements
show()	Shows the selected elements
slideDown()	Slides-down (shows) the selected elements
slideToggle()	Toggles between the slideUp() and slideDown() methods
slideUp()	Slides-up (hides) the selected elements
stop()	Stops the currently running animation for the selected elements
toggle()	Toggles between the hide() and show() methods

jQuery HTML / CSS Methods

The following table lists all the methods used to manipulate the HTML and CSS.
The methods below work for both HTML and XML documents. Exception: the html() method.

Method	Description
addClass()	Adds one or more class names to selected elements
after()	Inserts content after selected elements
append()	Inserts content at the end of selected elements
appendTo()	Inserts HTML elements at the end of selected elements
attr()	Sets or returns attributes/values of selected elements
before()	Inserts content before selected elements
clone()	Makes a copy of selected elements
css()	Sets or returns one or more style properties for selected elements
detach()	Removes selected elements (keeps data and events)
empty()	Removes all child nodes and content from selected elements
hasClass()	Checks if any of the selected elements have a specified class name
height()	Sets or returns the height of selected elements
html()	Sets or returns the content of selected elements
innerHeight()	Returns the height of an element (includes padding, but not border)
innerWidth()	Returns the width of an element (includes padding, but not border)
insertAfter()	Inserts HTML elements after selected elements
insertBefore()	Inserts HTML elements before selected elements
offset()	Sets or returns the offset coordinates for selected elements (relative to the document)
offsetParent()	Returns the first positioned parent element
outerHeight()	Returns the height of an element (includes padding and border)
outerWidth()	Returns the width of an element (includes padding and border)
position()	Returns the position (relative to the parent element) of an element
prepend()	Inserts content at the beginning of selected elements
prependTo()	Inserts HTML elements at the beginning of selected elements
prop()	Sets or returns properties/values of selected elements
remove()	Removes the selected elements (including data and events)
removeAttr()	Removes one or more attributes from selected elements
removeClass()	Removes one or more classes from selected elements
removeProp()	Removes a property set by the prop() method
replaceAll()	Replaces selected elements with new HTML elements
replaceWith()	Replaces selected elements with new content
scrollLeft()	Sets or returns the horizontal scrollbar position of selected elements
scrollTop()	Sets or returns the vertical scrollbar position of selected elements
text()	Sets or returns the text content of selected elements
toggleClass()	Toggles between adding/removing one or more classes from selected elements
unwrap()	Removes the parent element of the selected elements
val()	Sets or returns the value attribute of the selected elements (for form elements)
width()	Sets or returns the width of selected elements
wrap()	Wraps HTML element(s) around each selected element
wrapAll()	Wraps HTML element(s) around all selected elements
wrapInner()	Wraps HTML element(s) around the content of each selected element