Лабораторная работа № 11. Знакомство с jQuery

Цель: ознакомиться с возможностями jQuery для создания графических и визуальных эффектов и анимации.

Теория

jQuery – это библиотека, которая значительно упрощает и ускоряет написание JavaScript кода. jQuery позволяет создавать анимацию, обработчики событий, значительно облегчает выбор элементов в DOM. Данная библиотека работает со всеми браузерами.

Для jQuery написано огромное количество плагинов, которые позволяют расширить ее возможности еще больше. Плагин (plug-in) – это модуль к программе, который создается отдельно и, в случае необходимости, может быть подключен к уже работающему приложению.

Чтобы использовать jQuery, необходимо скачать ее с официального сайта (http://jquery.com) и добавить на страницу, вставив в секцию head следующий код:

```
<script type="text/javascript"
src="путь_к_скачанному_файлу/jquery.js"></script>
```

Также можно использовать jQuery удаленно. Для этого нужно добавить на страницу в секцию head следующий код:

```
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.
js"></script>
```

Код jQuery состоит из последовательно идущих команд. Стандартный синтаксис jQuery команд:

```
$(селектор).метод();
```

Знак $ сообщает, что символы, идущие после него, являются jQuery кодом; селектор позволяет выбрать элемент на странице; метод задает действие, которое необходимо совершить над выбранным элементом. Методы в jQuery разделяются на следующие группы:

- методы для манипулирования DOM;
- методы для оформления элементов;
- методы для создания AJAX запросов;
- методы для создания эффектов;
- методы для привязки обработчиков событий.

jQuery можно комбинировать с обычным JavaScript. Если строка начинается с $ – это jQuery, если $ в начале строки отсутствует – это строка JavaScript кода.

Селекторы jQuery

С помощью селекторов можно выбирать элементы на странице для применения к ним определенных действий.

Ниже располагается таблица с различными примерами использования селекторов для выбора желаемых элементов:

Пример	Результат
$("p")	Будут выбраны все элементы p, которые находятся на странице.
$(".par")	Будут выбраны все элементы на странице с class="par".
$("#par")	Будет выбран первый элемент на странице с id="par".
$(this)	Позволяет выбрать текущий HTML элемент. Щелкните на $(this) слева, чтобы посмотреть пример использования данного селектора в онлайн редакторе.
$("p.par")	Будут выбраны все элементы p на странице с class="par".
$("p#par")	Будут выбраны все элементы p на странице с id="par".
$(".par,.header,#heat")	Будут выбраны все элементы на странице со значениями атрибутов class="par", class="header" и id='heat'.
$("[src]")	Будут выбраны все элементы на странице, имеющие атрибут src.
$("[src='значение']")	Будут выбраны все элементы со значениям атрибута src="значение".
$("[src$='.gif']")	Будут выбраны все элементы со значениями атрибута src заканчивающимися на .gif.
$("div#wrap .par1")	Будут выбраны все элементы с class=par1, которые находятся внутри элементов div с id=wrap.
$(":input")	Будут выбраны все input элементы на странице.
$(":тип")	Будут выбраны все input элементы с <input type='тип' />. Например :button выберет все <input type='button' /> элементы, :text выберет все <input type='text' /> элементы и т.д.

Пример

```
$(document).ready(function(){
//Установим размер шрифта всех абзацев равным 20 пикселям
   $("p").css("fontSize","20px");
//Изменим на зеленый цвет шрифта элемента с id=el2
   $("#el2").css("color","green");
//Изменим на красный цвет шрифта элемента с class=el3
   $(".el3").css("color","red");
//Сделаем жирным шрифт элементов с id=el2 и class=el3
   $("#el2,.el3").css("fontWeight","bold");
```

```
//Изменим на синий цвет текста кнопки
   $(":input").css("color","blue");
//Установим размер шрифта всех элементов, имеющих атрибут href
равным 20 пикселям
   $("[href]").css("fontSize","20px");
//Изменим на зеленый цвет ссылки на www.belstu.by
$("[href='http://www.belstu.by/']").css("color","green");
});
```

Для сокращения размера кода можно соединять команды jQuery в цепочки. Команды в цепочке будут выполняться поочередно слева направо. Например,

```
<script type='text/javascript'>
//Код без сокращения
$("p").css("color","green");
$("p").css("font-size","30px");
//Сокращенный код
$("p").css("color","green").css("font-size","30px");
</script>
```

Обработчики событий jQuery

Обработчики событий – это функции, код которых исполняется только после совершения определенных действий.

Обработчики событий присутствовали и в JavaScript, но jQuery облегчает их использование и расширяет их функциональность.

Примеры действий, после которых выполняются обработчики:

- курсор мыши наведен на элемент;
- веб-страница или картинка полностью загружена;
- изменено содержимое поля формы;
- HTML форма отправлена;
- нажата клавиша на клавиатуре.

Общий вид определения обработчиков jQuery:

```
$(селектор).обработчик_события (function(){код_обработчика_события});
```

Код обработчика **mouseover** будет выполнен, когда курсор мыши будет наведен на элемент.

Код обработчика **mouseout** будет выполнен, когда курсор мыши будет выведен за границы элемента. Например,

```
$(document).ready(function(){
$("p").mouseover(function(){$("p").css("color","green")});
$("p").mouseout(function(){$("p").css("color","black")});
```

```
});
```

Код обработчика **click** будет выполнен после одинарного щелчка мыши на элементе.

Код обработчика **dblclick** будет выполнен после двойного щелчка мыши на элементе. Например,

```
    $(document).ready(function(){
       $("#but1").click(function(){alert("Вы нажали один раз на первую кнопку!");});
       $("#but2").dblclick(function(){alert("Вы нажали два раза на вторую кнопку!");});
    });
```

Код обработчика **focus()** будет выполнен, когда элемент станет активным.

Код обработчика **blur()** будет выполнен, когда элемент перестанет быть активным.

Код обработчика **change()** будет выполнен, при изменении содержимого элемента.

```
$(document).ready(function(){
   $("#el1").focus(function(){$(this).attr("value","")});
   $("#el1").blur(function(){$(this).attr("value","Введите ФИО")});
   $("#el2").change(function(){     alert("Содержимое     данного элемента было изменено.") });     });
```

С помощью jQuery можно также вызывать обработчики событий, привязанные к элементу.

Например: **$('#test').blur()** вызовет обработчик blur у элемента с id='test':

```
$(document).ready(function(){
// Зададим обработчик события, который будет выводить сообщение при нажатии на кнопку с id=but1
   $("#but1").click(function(){alert("Вы нажали на кнопку с id=but1")});
// Вызовем обработчик click у элемента с id='but1'
   $("#but1").click();
});
```

Эффекты jQuery

С помощью методов **fadeOut()**, **fadeIn()** и **fadeTo()** можно постепенно скрывать и отображать элементы анимированно.

Синтаксис:

```
//Позволяет постепенно скрыть выбранный элемент
$("селектор").fadeOut(скорость,функция);
//Позволяет постепенно отобразить выбранный элемент
```

```
$("селектор").fadeIn(скорость,функция);
//Позволяет постепенно скрыть/отобразить элемент до указанного значения прозрачности
$("селектор").fadeTo(скорость,прозрачность,функция);
```

Например,

```
$(document).ready(function(){
   $("#but1").click(function(){$("#par1").fadeOut(3000)});
   $("#but2").click(function(){$("#par1").fadeIn(3000)});
$("#but3").click(function(){$("#par1").fadeTo(3000,0.3)});
$("#but4").click(function(){$("#par1").fadeTo(3000,0.8)});
$("#but5").click(function(){$("#par1").fadeOut(3000,function(){
alert("Абзац был полностью скрыт.");});
   });
});
```

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

Синтаксис:

```
//Позволяет изменяет высоту элемента до 0
$("селектор").slideUp(скорость,функция);
//Позволяет плавно возвратить элементу его изначальную высоту
$("селектор").slideDown(скорость,функция);
//При первом вызове будет действовать как slideUp, а при втором как slideDown
$("селектор").slideToggle(скорость,функция);
```

Например,

```
$(document).ready(function(){
  $("#but1").click(function(){$("#square").slideUp(3000)});
$("#but2").click(function(){$("#square").slideDown(3000)});
$("#but3").click(function(){$("#square").slideToggle(3000)});
$("#but4").click(function(){$("#square").slideUp(3000,function(){
alert("Текст был скрыт");});
   });
});
```

С помощью метода **slideToggle** можно создавать на страницах удобные выпадающие меню. Например,

```
$(document).ready(function(){
$("#menu").click(function(){$("#list").slideToggle(2000)});
   $("#menu").toggle(function(){
      $("#img").attr("src","menudown.gif")},function(){
```

```
      $("#img").attr("src","menuup.gif")
   });
$("#menu").mouseover(function(){$("#menu").css("background-
color","#01939a")});
$("#menu").mouseout(function(){$("#menu").css("background-
color","#006064")});
});
```

Анимация в jQuery

С помощью метода **animate()** можно создавать полноценную анимацию.

Синтаксис:

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

где:

стили – CSS стили для анимации (к элементу одновременно может быть применено несколько стилей);

скорость – скорость анимации. Можно указать скорость с помощью предопределенных свойств: "slow", "fast", "normal" (медленно, быстро, нормально) или указать скорость в миллисекундах;

функция_смягчения – функция, которая будет отвечать за плавность выполнения анимации;

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

Пример:

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

Управление стилями в jQuery

jQuery имеет группу различных методов значительно упрощающих оформление элементов. Одним из самых важных методов в этой группе является метод **css()**.

С помощью метода **css** можно узнавать текущие или устанавливать новые значения свойств оформления элементов.

Синтаксис:

```
//Узнаем значение указанного CSS свойства выбранного элемента
$("селектор").css("свойство");
//Установим новое значение указанному CSS свойству элемента
```

```
$("селектор").css("свойство","значение");
//Установим произвольные значения нескольким CSS свойствам выбранного элемента
$("селектор").css({свойство1:значение1, свойствоN: значениеN});
```

Например,

```
$(document).ready(function(){
   $("#but1").click(function(){
      alert($("#par1").css("color"));
   });
   $("#but2").click(function(){
      $("#par2").css("color","red");
   });
   $("#but3").click(function(){
      $("#par3").css({"font-size":"27px","color":"red","font-family":"Arial"});
   });
   $("#text1").focus(function(){
      $("#text1").val("");
   });
   $("#but4").click(function(){
      $("#par4").css("font-size",$("#text1").val()+"px");
   });
});
```

Задания к лабораторной работе № 10

Задание 1. Выполните задания предыдущих лабораторных работ №8 и № 9 с использованием jQuery.