УДК 004.655, 004.657, 004.62

Т.В. Зудилова, М.Л. Буркова

Web-программирование JavaScript - СПб: НИУ ИТМО, 2012. – 68 с.

В пособии излагаются методические рекомендации к выполнению лабораторных работ по дисциплине “Web-программирование ”.

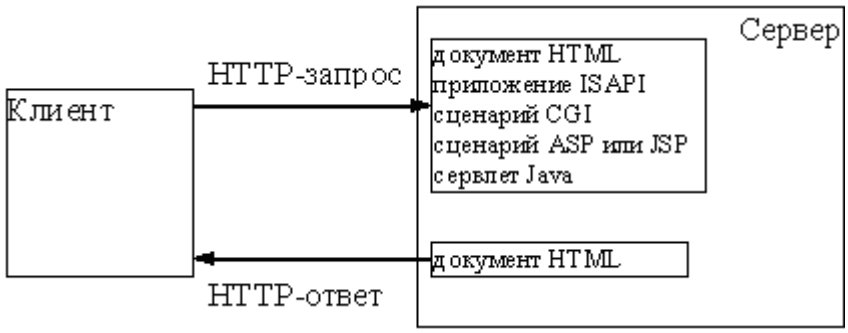
Предназначено для студентов, обучающихся по всем профилям подготовки бакалавров направления: 210700 Инфокоммуникационные технологии и системы связи.

Рекомендовано к печати Ученым советом факультета инфокоммуникационных технологий, протокол №4 от 13 декабря 2011г.

В 2009 году Университет стал победителем многоэтапного конкурса, в результате которого определены 12 ведущих университетов России, которым присвоена категория «Национальный исследовательский университет». Министерством образования и науки Российской Федерации была утверждена программа его развития на 2009–2018 годы. В 2011 году Университет получил наименование «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

1. Обзор возможностей языка JavaScript

Взаимодействие клиента и сервера в Интернете осуществляется с помощью запросов, посылаемых клиентом серверу, и ответов сервера на запрос клиента:



Его основу составляют HTTP-сообщения, подразделяемые на:

- запрос (request) клиента к серверу;
- ответ (response) сервера клиенту.

Стандартный язык разметки HTML позволяет легко создавать статичные Web–страницы. Пользователь не может менять их содержимое, не может взаимодействовать с ними. Для того чтобы сделать страницу по-настоящему интерактивной, нужен еще один язык, выполняемый в контексте браузера, - скриптовый язык.

Исследования работы приложений интернета показали, что для выполнения определенных действий пользователя нет необходимости постоянно обращаться к серверу - эти действия можно реализовать на стороне клиента, если бы он позволял каким-то образом их запрограммировать. Так появился встроенный в программу просмотра Web-страниц (браузер) язык JavaScript, который расширил возможности языка разметки HTML, предоставляя разработчику возможность встраивать в документ HTML код программы, выполняющейся на клиенте.

Скриптовый язык используется для создания интерактивных страниц. Обычно он не содержит всех возможностей настоящих языков программирования, таких, например, как работа с файлами или управление графикой. Созданные с помощью скриптовых языков программы не могут выполняться самостоятельно - они работают только в контексте браузера, поддерживающего выполнения скриптовых программ. Создаваемые на скриптовых языках программы, называются сценариями или скриптами, включаются в состав Web-страниц и распознаются и обрабатываются браузером отдельно от остального HTML - кода.

Язык программирования JavaScript - объектно-ориентированный язык разработки встраиваемых приложений, выполняющихся как на стороне клиента, так и на стороне сервера.

Веб-обозреватель, работающий на компьютере-клиенте, обеспечивает среду, в которой JavaScript имеет доступ к объектам, которые представляют собой окна, меню, диалоги, текстовые области и т. д. Кроме того, обозреватель позволяет присоединить сценарии на языке JavaScript к

1. Объекты – это сложные сущности, позволяющие хранить сразу несколько значений разных типов данных, они представляют собой блоки, из которых строится JavaScript. Применяются для возвращения значений и изменения состояния форм, страниц, браузера и определенных программистом переменных. Объекты можно сопоставить с существительными. Кошка, автомобиль, дом, компьютер, форма – все это существительные, они могут быть представлены как объекты.
2. Экземпляры объекта – сущности, хранящие реальные данные и созданные на основе этого объекта. То есть конкретный, реально существующий дом, находящийся по заданному адресу можно рассматирвать, как экземпляр объекта типа дом.
3. Свойства – набор внутренних параметров объекта. Используются для того, чтобы различать экземпляры одного объекта – например, все экземпляры типа дом. Свойства сравнимы с прилагательными и ссылаются на уникальные для каждого экземпляра объекта особенности. Один и тот же объект может обладать многими свойствами: дом может быть большим и маленьким, синим и красным. Разные объекты могут обладать одинаковыми свойствами: дерево, так же, как и дом, может быть большим и маленьким, синим и красным... Большинство свойств объекта мы можем изменять, воздействуя на них через методы.
4. Методы - это действие или способ, при помощи которого мы можем изменять определенные свойства объекта, то есть управлять этими объектами, а также в некоторых случаях менять их содержимое.
5. События – это очень важное в программировании на JavaScript понятие. События главным образом порождаются пользователем, являются следствиями его действий. Если пользователь нажимает кнопку мыши, то происходит событие, которое называется Click. Если экранный указатель мыши движется по ссылке HTML-документа, происходит событие MouseOver. Существует несколько различных событий.
6. Оператор - это команда, инструкция для компьютера. Встретив в программе тот или иной оператор, машина четко его выполняет.
7. Функция - это определенная последовательность операторов, то есть набор команд, последовательное выполнение которых приводит к какому-то результату. Например, выполнение кем-то заданной Вами функции (процедуры) "возьми стакан, открой кран, набери в него воды и принеси мне" приведет к результату: Вы получите стакан воды из-под крана.
8. Переменная - в языках программирования переменные используются для хранения данных определенного типа, например параметров свойств объекта. Каждая переменная имеет свое имя (идентификатор) и хранит только одно значение, которое может

Параметр language задает используемый язык сценариев. В случае языка JavaScript его значение задавать не обязательно, так как этот язык используется браузерами по умолчанию.

Примечание:
символы // перед закрывающим тэгом комментария --> являются оператором комментария JavaScript. Он необходим для правильной работы интерпретатора.

Документ может содержать несколько тэгов <script>, расположенных в любом месте документа. Все они последовательно обрабатываются интерпретатором JavaScript по мере отображения частей документа в окне браузера. В связи с этим ссылка на переменную, определенную в сценарии, размещенном в конце документа, может привести к генерации ошибки интерпретатора при обращении к такой переменной из сценария в начале документа.

2. Задание файла с кодом JavaScript.

Тэг <script> имеет параметр src, позволяющий связать встраиваемый сценарий с внешним файлом, содержащим программный код на языке JavaScript. В качестве значения параметра задается полный или относительный URL-адрес ресурса. Задание закрывающего тэга </script> обязательно, независимо от того, заданы или нет операторы внутри тэга. Следующий фрагмент кода связывает документ HTML с файлом-источником, содержащим некоторый набор функций:

```
<script language="JavaScript" src="http://url/file.js">
  операторы javascript
</script>
```

Примечание:
связываемый внешний файл не должен содержать тэгов HTML и должен иметь расширение .js.

3. Использование выражений JavaScript в качестве значений параметров тэгов HTML.

Переменные и выражения JavaScript можно использовать в качестве значений ьпараметров тэгов HTML. Например:

```
<a href="javascript: window.open('name.htm', '_self')">
<img src="images/ch1.png" border="0">
</a>
```

4. Определение обработчика событий в тэге HTML.

1.2. Язык ядра JavaScript

Синтаксис языка

Язык JavaScript чувствителен к регистру.

\'	Апостроф
\"	Двойные кавычки
\\	Обратная наклонная черта

ESC-последовательности форматирования используются при отображении информации в диалоговых окнах, отображаемых функциями alert(), prompt() и confirm(), а также, если методом document.write() записывается содержимое элемента pre.

Комментарии в программе JavaScript двух видов - однострочные и многострочные:

```
// комментарий, расположенный на одной строке.
/*
  комментарий, расположенный
  на нескольких строках.
*/
```

Ссылка на объект осуществляется по имени, заданному параметром name тэга HTML, с использованием точечной нотации. Например, пусть в документе задана форма с двумя полями ввода:

```
<form name="form1">
Фамилия: <input type = "text" name = "student" size = 20>
Курс: <input type = "text" name = "course" size = 2>
</form>
```

Для получения фамилии студента, введенного в первом поле ввода, в программе JavaScript следует использовать ссылку document.form.student.value, а чтобы определить курс, на котором обучается студент, необходимо использовать ссылку document.form.course.value.

Переменные и литералы в JavaScript

В JavaScript все переменные вводятся с помощью одного ключевого слова var. Синтаксическая конструкция для ввода в программе новой переменной с именем name1 выглядит следующим образом:

```
var name1;
```

Объявленная таким образом переменная name1 имеет значение 'undefined' до тех пор, пока ей не будет присвоено какое-либо другое значение, которое можно присвоить и при ее объявлении:

```
var  name1 = 5;
var  name1 = "новая строковая переменная";
```

JavaScript поддерживает четыре простых типа данных:

- Целый
- Вещественный
- Строковый
- Логический (булевый)

<=	Меньше или равно		
>	Строго больше		
<	Строго меньше		

4. Строковые выражения

Вычисляемым значением строкового выражения является число. В JavaScript существует только один строковый оператор – оператор конкатенации (сложения) строк:

string1 = “Моя ” + “строка”

1.3. Управляющие конструкции языка JavaScript

Операторы JavaScript

Операторы служат для управления потоком команд в JavaScript. Блоки операторов должны быть заключены в фигурные скобки.

1. Операторы выбора

- условный оператор if

Эта управляющая структура используется, когда необходимо выполнить некий программный код в зависимости от определенных условий. Также предусмотрена конструкция if-else (если-тогда-иначе).

```
if (условие_1)
{
 оператор_1;          // эти операторы выполняются, если условие_1 верно
 оператор_2;
 }
else
{
оператор_3;   // эти операторы выполняются, если условие_1 ложно
оператор_4;
 }
```

Условие для проверки (вопрос компьютеру) записывается сразу после слова if в круглых скобках. После этого в фигурных скобках пишется то, что будет предприниматься в случае выполнения условия. Далее else и снова в фигурных скобках то, что выполнится в случае, если условие не сработает. Количество различных действий между фигурными скобками неограниченно, фактически можно выполнить две различные программы. При сравнении можно использовать логические выражения. Например:

случае равно единице). Это выражение инициализации выполняется самым первым и всего один раз.

Второй параметр (i<10) - это условие, которое должно быть истинным, чтобы цикл выполнялся, как только условие цикла становится ложным, работа цикла завершается. Он называется условием цикла. Проверка условия цикла осуществляется на каждом шаге; если условие истинно, то выполняется тело цикла (операторы в теле цикла). Цикл в данном случае выполнится только девять раз так как задано условие i<10.

Третий параметр (i++) - это оператор, который выполняется при каждом последовательном прохождении цикла. Он называется выражением инкремента, поскольку в нем задается приращение счетчика (приращение счетчика в данном случае равно единице). Пример автоматической прорисовки нескольких линий с помощью цикла for.

```
<script language="JavaScript" type="text/JavaScript">
for (var i=1; i<10; i++){
   document.write("<hr align='center' width='100'>");
}
</script>
```

- цикл while (цикл с предусловием)

```
while (условие)
{
   <тело цикла>
}
```

Пока значение условия - true (истинно), выполняется тело цикла. Тело цикла может быть представлено простым или составным оператором.

Оператор while содержит в скобках все необходимые параметры условия цикла (логическое выражение). После определения всех параметров цикла вводится открывающая фигурная скобка, символизирующая начало тела цикла. Закрывающая фигурная скобка вводится в конце тела цикла. Все операторы, введенные в скобках, выполняются при каждом прохождении цикла.

```
<script language="JavaScript">
 var i=1;
 while(i<=10){
 document.write('число='+i+'<br>');
 i=i+2;
 }
</script>
```

- прерывание и перезапуск цикла

Оператор прерывания break позволяет прервать выполнение цикла и перейти к слудующему за ним выражению:

```
a = 10;
i = 1;
while (a<100){
```

```
function myMessage()
{
alert("My Message")
}
</script>
<body onload='setTimeout ("myMessage()",3000)'>
<p>Каждые три секунды будет появляться сообщение</p>
</body>
```

Метод setTimeout() запускает выполнение кода JavaScript, задаваемого первым строковым параметром, через определенный промежуток времени после выполнения метода.

Интервал задается в миллисекундах (1000 соответствует 1 секунде).

1.4. Стандартные объекты и функции ядра JavaScript

Объект Array

Массив - упорядоченный набор однородных данных, к элементам которого можно обращаться по имени и индексу. Язык JavaScript не имеет встроенного типа данных для создания массивов, поэтому для решения используется объект Array и его методы.

Для создания объекта Array вызывается оператор new и конструктор массива - системная функция (ее имя совпадает с именем объекта), инициализирующая элементы массива:

```
m=new Array();
```

Заполнение массива происходит позже. Например:

```
<script language="JavaScript">
 //создание нового массива
m=new Array();
 //заполнение массива
m[0]=1;
m[1]=2;
m[2]=4;
m[3]=56;
</script>
```

В приведенном выше примере с помощью команды new создается массив m, а затем происходит его заполнение - каждому элементу присваивается определенное значение.

```
m=new Array(1,2,4,56);
```

Вызывается команда new и сразу задаются значения всех элементов массива.

```
<script language="JavaScript">
 //создание нового массива и его заполнение
```

Данный объект создается также, как и любой объект в JavaScript – с помощью оператора new и конструктора, в данном случае Date():

```
date1 = new Date();  // значением переменной date1 будет текущая дата
```

Параметром конструктора может быть строка, в которой записана нужная дата:

```
date1 = new Date("january 14, 2000, 12:00:00");
```

Можно задать список параметров:

```
date1 = new Date(2000, 1, 14, 12, 0, 0);
```

Объект Math

В свойствах данного объекта хранятся основные математические константы, а его методы вычисляют основные математические функции. При обращении к данному объекту, создавать его не надо, но необходимо явно указывать его имя Math. Например:

```
p = Math.PI;   // хранится значение числа пи.
```

Объект String

Можно явно создавать строковый объект, используя оператор new и конструктор:

```
myString = new String("Hello!");
```

Данный объект имеет единственное свойство length, хранящее длину строки, содержащейся в строковом объекте, и два типа методов: одни непосредственно влияют на содержание самой строки, вторые возвращают отформатированный HTML-вариант строки.

Стандартные функции верхнего уровня

В JavaScript существуют несколько функций, для вызова которых не надо создавать никакого объекта, она находятся вне иерархии объектов.

Функция parseFloat(parameter) анализирует значение переданного ей строкового параметра на соответствие представлению вещественного числа.

Функция parseInt(parameter, base) пытается возвратить целое число по основанию, заданному вторым параметром.

Эти функции полезны при анализе введенных пользователем данных в полях формы до их передачи на сервер.

Функции Number(object) и String(object) преобразуют объект, заданный в качестве его параметра в число или строку.

свойства позволяют определить характеристики программы просмотра. Каждая страница в добавление к объекту navigator обязательно имеет еще четыре объекта:

- window — объект верхнего уровня, свойства которого применяются ко всему окну, в котором отображается документ;
- document — свойства которого определяются содержимым самого документа: связи, цвет фона, формы и т. д.;
- location — свойства которого связаны с url-адресом отображаемого документа;
- history — представляет адреса ранее загружавшихся HTML-страниц.

Кроме указанных объектов страница может иметь дополнительные объекты, зависящие от ее содержимого, которые являются дочерними объектами объекта document. Если на странице расположена форма, то все ее элементы являются дочерними объектами этой формы. Для задания точного имени объекта используется точечная нотация с полным указанием всей цепочки наследования объекта. Наиболее общий объект высшего уровня находится слева в выражении, и слева направо происходит переход к более частным объектам, являющимся при этом наследниками высших в иерархии объектов.

Кроме этих классов объектов пользователь может создавать и свои собственные. Но обычно большинство программ используют эту систему классов и не создают новых.

Объект navigator

Этот объект применяется для получения информации о версиях.

Синтаксис:

navigator.name_properties

Методы и события, догадаться не определены для этого объекта. Да и свойства только для чтения, так как ресурс с информацией о версии недоступен для редактирования.

Свойства

- appCodeName - кодовое имя браузера;
- appName - название браузера;
- appVersion - информация о версии браузера;
- userAgent - кодовое имя и версия браузера;

Ниже приведен пример использования объекта navigator.

```
<html>
<head>
<title> navigator </title>
<script language="javascript">
var firstn = window.prompt("Введите ваше имя: ","ваше имя")
function welcome(){
```

Таблица 5.

Параметр	Значение		Описание
fullscreen	yes	no	указывает, показывается ли новое окно на полный экран или как обычное окно. По умолчанию показывается обычное окно
	1	0	
channelmode	yes	no	позволяет указать, отображается ли полоса каналов
	1	0	
toolbar	yes	no	позволяет указать, отображается ли полоса кнопок
	1	0	
location	yes	no	позволяет указать, отображается ли полоса для ввода адреса
	1	0	
directories	yes	no	позволяет указать, отображается ли полоса кнопок для выбора каталогов
	1	0	
status	yes	no	позволяет указать, отображается ли полоса статуса
	1	0	
menubar	yes	no	позволяет указать, отображается ли полоса меню
	1	0	
scrollbars	yes	no	задает отображение горизонтальной и вертикальной полос прокрутки
	1	0	
resizable	yes	no	позволяет указать, может ли окно изменять свой размер
	1	0	
width	yes	no	задает ширину окна в пикселах. Минимальное значение - 100
	1	0	
height	yes	no	задает высоту окна в пикселах. Минимальное значение - 100
	1	0	
top	yes	no	задает вертикальную координату левого верхнего угла окна
	1	0	
left	yes	no	задает горизонтальную координату левого верхнего угла окна
	1	0	

	http://www.someplace.com, то свойством referrer будет: http://www.someplace.com, если это свойство было в странице вышеупомянутого расположения; в противном случае оно обращается в null
location	Соответствует адресу url текущего документа

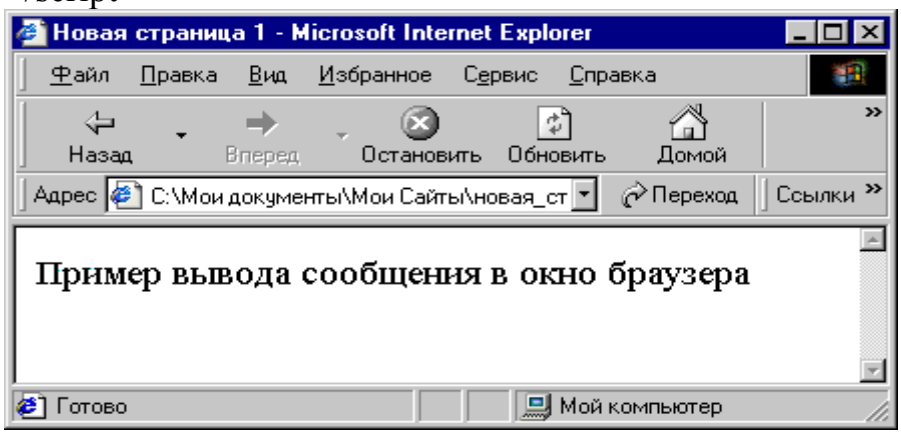
Таблица 7.

Методы	Назначение
write() writeln()	Записывает HTML-текст в текущий документ и должен вызываться, когда документ открывается для записи. Синтаксис: document.write('somestring'), где somestring может быть одной строкой, переменной или же несколькими связанными строками в формате HTML, которые выводятся на экран
lastModified()	Показывает дату последней модификации документа: date1 = document. lastModified
open()	Открывает документ для записи дополнительных строк в формате HTML: document.open()
close()	Закрывает документ, который вызывался методом document.open(): document.close().

Методы write() / writeln().

Вызов метода document.write() с указанием определенных параметров приводит к отображению текста в окне браузера. В качестве параметра при вызове метода document.write() мы указываем строку, которую хотели бы увидеть на экране.

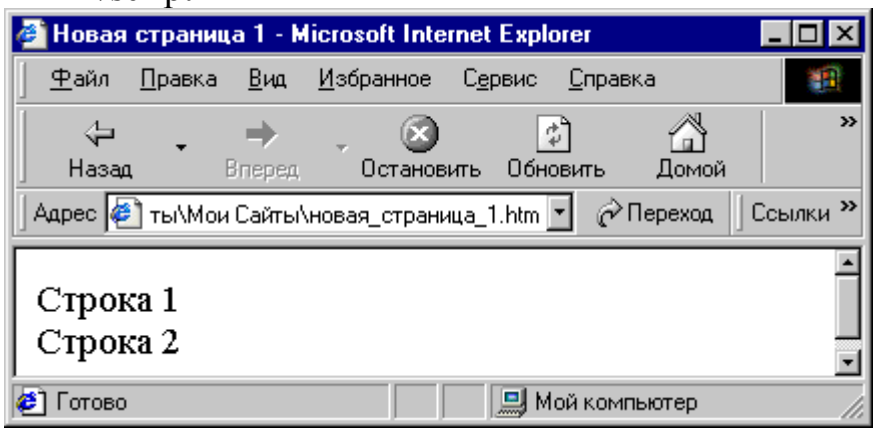
```
<script language="JavaScript">
 document.write('Пример вывода сообщения в окно браузера')
</script>
```



Выводимая строка может содержать и тэги языка HTML. В этом случае браузер выведет данную строку точно так же, как если бы она была размещена непосредственно в HTML документе.

и

```
<script language="JavaScript">
document.writeln('Строка 1') ;
document.writeln('Строка 2') ;
</script>
```



Объект location

Объект location содержит информацию о местонахождении текущего документа, т.е. его интернет-адрес. Его также можно использовать для перехода на другой документ и перезагрузки текущего документа.

Таблица 8.

Свойство	Описание
hash	Имя "якоря" в интернет-адресе документа, если оно есть.
host	Имя компьютера в сети интернет вместе с номером порта, если он указан.
hostname	Имя компьютера в сети Интернет.
href	Полный интернет-адрес документа.
pathname	Путь и имя файла, если они есть.
port	Номер порта. Если не указан, возвращает номер 80 - стандартный порт, через который работает протокол http.
protocol	Идентификатор протокола. Если не указан, возвращается "http:".
search	Строка параметров, если она есть.

document.form1.text1;

Кроме имени элементы формы, имеют свойство value, значение которого определяется смыслом атрибута value элемента формы. Например, для элементов text и textarea значением этого свойства будет строка содержимого полей ввода этих элементов; для кнопки подтверждения - надпись на кнопке и т.д. Обратиться к свойству value можно по тому же принципу, например:

document.form1.text1.value

1.6. Обработка событий

Использование языка JavaScript при обработке событий значительно расширило возможности языка HTML. Чаще всего программы создаются для обработки информации, вводимой пользователем в поля форм. Возможности управления элементами форм обеспечиваются главным образом за счет функций обработки событий, которые могут быть заданы для всех элементов формы. События делятся на несколько категорий:

- события, связанные с документами (события документа) - загрузка и выгрузка документов;
- события, связанные с гиперсвязью (события гиперсвязи) - помещение указателя мыши на гиперсвязь и активизация гиперсвязи;
- события, связанные с формой (события формы) –
    - щелчки мыши на кнопках;
    - получение и потеря фокуса ввода и изменение содержимого полей ввода, областей текста и списков;
    - выделение текста в полях ввода и областях текста;

События, связанные с документами, возникают при загрузке и выгрузке документа, в то время как события гиперсвязей возникают при их активизации или при помещении на них указателя мыши. Чтобы обеспечить перехват события, необходимо написать функцию-обработчик события. В качестве обработчиков событий могут быть заданы целые функции языка JavaScript или только группы из одного или нескольких операторов. В таблице перечислены имена событий и условия их возникновения:

Таблица 10.

Имя события	Атрибут HTML	Условие возникновения события
Blur	onBlur	Потеря фокуса ввода элементом формы
Change	onChange	Изменение содержимого поля ввода или области текста, либо выбор нового элемента списка

```
<input type="button" value="Вторая кнопка" onClick="but2()">
</form>
…
```

Работа с меню

Список в форме задается с помощью объекта select, обработка событий выполняется с помощью следующих параметров: onChange - вызывается при изменении выбора; onBlur - вызывается при снятии фокуса с объекта; onFocus - вызывается при перемещении фокуса на объект.

Рассмотрим следующий пример:

```
…
<script language="JavaScript">
function selectBlur()
{
   document.myForm7.myText.value="Вы нажали поле вне списка ";
}
function selectFocus()
{
   document.myForm7.myText.value="Вы нажали ту же кнопку ";
}
function selectChange()
{
   document.myForm7.myText.value="Вы нажали другую кнопку ";
}
</script>
…
<form name="myForm7">
<input type="text" name="myText" size=40 value="Город"><br>
<select name="script" multiple onBlur="selectBlur()"
onFocus="selectFocus()" onChange="selectChange()">
<option value="town1" selected>Париж
<option value="town2">Лондон
<option value="town3">Рим
<option value="town4">Берлин
</select>
</form>
…
```

Управление логикой программного кода при помощи событий

В объектно-ориентированном программировании нет единой структуры управления работой программы. Есть независимые друг от

пересылки их на сервер. В следующем примере разъясняется, как выполнять эту процедуру.

Рассмотрим скрипт, который будет проверять правильность заполнения формы. Необходимо проверить нет ли пустых строк и правильно ли введен e-mail:

```
<html>
<head>
<title>пример формы</title>
<script language="JavaScript">
    function doSend(){
    var v=document.user.e.value.indexOf("@",1)
    if(document.user.f.value==""){
    alert('Вы должны заполнить поле ФИО')
    document.user.f.focus()
     }
    if(document.user.a.value==""){
    alert('Вы должны заполнить поле адреса')
    document.user.a.focus()
    }
    if(document.user.e.value==""){
    alert('Вы должны заполнить поле e-mail')
    document.user.e.focus()
    }
   if(v==-1){
   alert('Адрес e-mail указан неверно')
   document.user.e.select()
   document.user.e.focus()
   }
   else
   document.user.submit()
   }
   </script>
</head>
<body>
<p align="center"><font size=6>Данные о пользователе</font>
<form name="user">
<b>Пожалуйста, укажите данные о себе:</b>
<br>
ФИО<input type="text" name="f" size="30"><br>
Адрес<input type="text" name="a" size="35"><br>
e-mai<input type="text" name="e" size="30"><br>
<input type="button" value="Послать" onClick="doSend()">
<input type="reset" value="Отменить">
</form>
```

истечения заданного промежутка времени. Изменяться может не только стиль и цвет текста, но и весь объект, текст или рисунок.

Объединение JavaScript и CSS

1. Пример изменения цвета текста.

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<h1 style="color:red">Доброе пожаловать на нашу страницу!</h1>
<p>Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации. </p>
    </body>
</html>
```

Данный пример применения CSS позволяет сделать заголовок красного цвета. Допустим, вы хотите, чтобы текст заголовка только тогда становился красным, когда пользователь наводит на него курсор. Этого можно добиться с помощью CSS и JavaScript.

Шаг 1. Удаление существующей информации о стиле

Это действие может показаться вам шагом назад, но оно действительно необходимо:

```
<html>
<head>
    <title>Простая страница</title>
</head>
<body>
<h1>Добро пожаловать на нашу страницу! </h1>
<p>Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации.
    Здесь много интересной информации. </p>
</body>
</html>
```

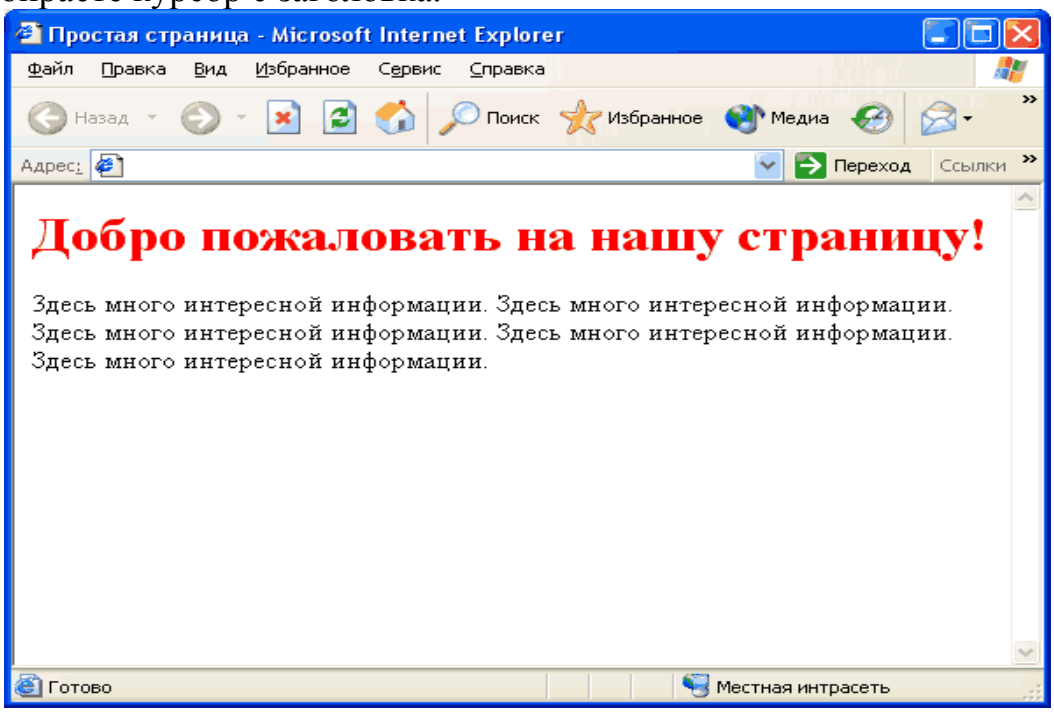
Шаг 2. Добавление идентификатора

Поскольку вам нужно как-то обращаться к элементу, с которым будут производиться манипуляции, необходимо в тэг <hl> добавить атрибут id - это краткое обозначение, позволяющее указать нужный элемент:

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function colorchange()
{
head1.style.color = "red";
}
</script>
</head>
<body>
<h1 id="head1" onMouseover="colorchange()">Добро пожаловать на
нашу страницу!</h1>
<p>3десь много интересной информации.
        3десь много интересной информации.
        3десь много интересной информации.
        3десь много интересной информации.
        3десь много интересной информации. </p>
</body>
</html>
```

Откройте эту страницу в браузере и посмотрите, что происходит, когда вы наводите курсор на заголовок. Если все было сделано правильно, то цвет заголовка изменится.

Но обратите внимание, что цвет текста не становится прежним, когда вы убираете курсор с заголовка.



2. Пример использования атрибута text-decoration.

элемент страницы, если бы не было представлено никакой информации о местоположении;

- top - используется для указания вертикального смещения элемента от точки отсчета. Величина смещения может выражаться в различных единицах (пикселы, дюймы, сантиметры, миллиметры и т.п.). В наших примерах используются пикселы. Положительное значение top соответствует смещению элемента страницы вниз, в то время как отрицательное - по направлению к верхней границе окна браузера;
- left - подобен атрибуту top, но применяется для указания горизонтального направления. Положительное значение соответствует сдвигу элемента вправо, отрицательное - влево.

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<h1 style=" text-decoration: underline">Добро пожаловать на нашу
страницу!</h1>
<p style="position:absolute; top:125px; left:200px">Простой текст для
позиционирования.</p>
</body>
</html>
```

С помощью CSS текст расположен со значением absolute, то есть его положение отсчитывается от верхнего левого угла окна браузера. Значение атрибута top равно 125px, таким образом, текст будет расположен на 125 пикселов ниже верхнего края страницы. Значение атрибута left равно 200px, то есть текст будет сдвинут на 200 пикселов от левого края окна браузера.

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

Затем воспользуемся CSS, чтобы поместить текст в начальное положение:

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<div id="anim" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

Далее начинаем работать над сценарием JavaScript. Поскольку не нужно, чтобы текст вечно двигался вправо, надо предусмотреть возможность контролирования этого процесса. Чтобы запустить сценарий на выполнение только при условии, если текст находится, например, менее чем в 500 пикселах от левой границы экрана, удобнее всего воспользоваться оператором if. Для этого понадобится атрибут CSS pixelLeft.

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anim.style.pixelLeft < 500)
{
}
}
</script>
</head>
<body>
<div id="anil" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

Теперь рассмотрим операторы, управляющие анимацией. Прежде всего нужно задать скачок. Каждый раз текст будет перемещаться вправо на 50 пикселей. Атрибут pixelLeft используется не только для определения положения текста, но и для изменения положения на 50 пикселей:

Процесс будет повторяться до тех пор, пока условие оператора if не станет ложным. Последнее, что нужно сделать, - запустить сценарий на выполнение. Для этого следует воспользоваться событием onLoad:

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anil.style.pixelLeft < 500)
{
anil.style.pixelLeft +=2;
setTimeout("moveTxt()", 50);
}
}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anil" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

2. Пример движения текста по диагонали

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anim.style.pixelTop < 500)
{
anim.style.pixelTop + = 2;
anim.style.pixelLeft +=2;
setTimeout("moveTxt()", 50);
}
}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anim" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
```

```
Слой 2 наверху
<div style="position:relative; font-size:50px; z-index:3; color: navy">
Слой 1
</div>
<div style="position:relative; top:-55; left:5; color:orange; font-size:80px;
z-index:4">
Слой 2
</div>
</body>
</html>
```

Тип позиционирования слоя определяется параметром position, положение элемента - двумя координатами top и left.

Кроме тегов и абсолютное позиционирование поддерживают следующие элементы: <applet>, <input>, <button>, <object>, <select>, <fieldset>, <iframe>, <table>, <img>, <textarea>.

Параметр position:relative используется для смещения слоя относительно родительского элемента. Установка этого значения не изменяет размещение элемента, но если установлены значения свойств top или left, то слой смещается от своего нормального положения в документе.

В то время как свойство position указывает тип системы координат, параметры top и left определяют точную позицию слоя. Значения этих параметров могут определяться в процентном отношении или пикселах, принимать положительные и отрицательные величины. Это дает возможность размещать контент выше или ниже на странице независимо от физической позиции кода HTML. То есть, в верхней части страницы можно поместить слой, который описан внизу HTML-документа.

Свойство z-index

Свойство z-index определяет порядок слоев, или их перекрытие по отношению к другим слоям. По умолчанию все слои позиционированы со значением z-index равным нулю. Другие слои могут размещаться ниже путем установки отрицательного значения z-index. Для слоев, у которых z-index не установлен, это значение назначается неявно в соответствии с их положением в документе. Поэтому слой, который помещен в документ позже, размещается выше остальных элементов, позиционированных ранее.

ссылку. Эту идею можно применить и для организации выпадающих меню.

```
<html>
<head>
<style type="text/css">
div {
position: absolute;
top: 20;
left: 160;
visibility: hidden;
}
</style>
<script language="JavaScript">
function show_d(d)
{
div1.style.visibility='hidden';
div2.style.visibility='hidden';
div3.style.visibility='hidden';
div4.style.visibility='hidden';
div5.style.visibility='hidden';
d.style.visibility='visible';
}
</script>
</head>
<body>
<a href="javascript:void(0)" onClick="show_d(div1);">
показать слой 1
</a><br>
<a href="javascript:void(0)" onClick="show_d(div2);">
показать слой 2
</a><br>
<a href="javascript:void(0)" onClick="show_d(div3);">
показать слой 3
</a><br>
<a href="javascript:void(0)" onClick="show_d(div4);">
показать слой 4
</a><br>
<a href="javascript:void(0)" onClick="show_d(div5);">
показать слой 5
</a><br>
<div id="div1">
<h3>Слой номер один</h3>
```

Некоторый текст, на слое расположенный. Его можно скрыть и показать. Текст может содержать несколько строк.

</tr>
</table>
…

- сохраните файл с именем main.js следующим образом: укажите тип файла “Все файлы”, кодировку “UTF-8”.

4. Добавьте ссылку на внешний скриптовый файл из рабочего HTML-документа.
5. Сохраните документ с именем Ex2.html в рабочей папке.

Задание 3.

1. Создайте простой HTML-документ.
2. Сохраните документ с именем Ex3.html в рабочей папке**.**
3. Добавьте в документ код JavaScript так, чтобы в диалоговом окне появлялось поле с надписью "Введите сюда своё имя" и со значением по умолчанию в поле "Введите имя". Для этого используйте метод promt(...) объекта window. Для хранения введенного значения заведите новую переменную.
4. Организуйте вывод введенного значения имени в окно браузера в виде: "Ваше имя <…..>".
5. Дополните код, чтобы в новом диалоговом окне появилось надпись "Начать заново? " При положительном ответе появлялось диалоговое окно: "Не надоело? ", при отказе – "Ну и правильно!". Используйте для написания методы alert(…) и confirm(…) объекта window.

2.2. Практическая работа №2. Операторы управления, функции. Объекты ядра JavaScript.

Задание 4.

1. Рассмотрите пример скрипта:

```
<html>
<head>
<title>if</title>
</head>
<body>
<script language="JavaScript" type="text/JavaScript">
var x, y;
x=parseInt(prompt("Введите значение x",""));      // метод parseInt() переводит строку в целое
y=parseInt(prompt("Введите значение y",""));   // число
if(x<y)
{
alert("Максимальное число - y")
}
else {
alert("Максимальное число - x")
```

3. Добавьте в документ код JavaScript так, чтобы в окне браузера была выведена таблица степеней двойки вида:

Степень	Результат
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32

Для этого в сценарии используйте метод write(…) объекта document для формирования содержимого страницы. На каждой итерации цикла for сформируйте очередную строку таблицы, в первую ячейку которой заносится соответствующая степень двойки, а во вторую результат ее возведения в указанную степень. Для выполнения этого действия используется встроенный объект Math и его метод pow(…), возводящий первый параметр в степень, заданную вторым параметром. Обратите внимание, что метод write(…) может вызываться с любым количеством фактических параметров. Результатом его работы в любом случае является вывод в документ строки, полученной конкатенацией всех параметров, переданных в метод.

Задание 7.

1. Рассмотрите пример скрипта:

```
<html>
<head>
<title>array</title>
</head>
<body>
<script language="JavaScript">
year=new Array("декабрь","январь","февраль","март","апрель","май",
"июнь","июль","август","сентябрь","октябрь","ноябрь");
summer=new Array();  //летние месяцы
summer=year.slice(6,9);
document.write(summer+"<br>");
</script>
</body>
</html>
```

2. Создайте массив, содержащий названия школьных предметов. Выделите из него два массива. Пусть к первому относятся предметы из раздела точных наук, а ко второму - из раздела гуманитарных

- внизу выводилась дата последней модификации документа, используйте для этого слияние методов wtite(…) и lastModified(…) объекта document.

3. Сохраните документ с именем Ex9.html в рабочей папке.

Задание 10.

1. Рассмотрите пример скрипта открытия нового окна на странице:

```
<html>
<head>
<title>window</title>
</head>
<body>
<h1>Создание нового окна</h1>
<hr>
<script language="JavaScript" type="text/JavaScript">
window.open("http://www.google.com","","toolbar=no,scrollbars=yes,width=250, height=250, resizable=yes, top=100, left=500")
</script>
</body>
</html>
```

2. Измените скрипт так, чтобы выполнялись следующие условия:
   - открытие нового окна происходило при нажатии на ссылку с текстом: «Щелкните на ссылке для получения справочной информации»,
   - размеры окна - 500x500,
   - есть возможность изменения размеров окна.

   Для выполнения задания используйте написание функции.
3. Сохраните документ с именем Ex10.html в рабочей папке.

Задание 11.

1. Создайте страницу с переадресацией на другой адрес (redirect).
2. Изменитескрипт так, чтобы переадресация на другой адрес была с задержкой 5 секунд.
3. Сохраните документ с именем Ex11.html в рабочей папке.

Задание 12.

1. Создайте HTML-документ, в котором будет 2 ссылки:
   - первая ссылка должна ссылаться на PDF файл; при нажатии на нее выпадает сообщение с предупреждением о том, что для загрузки документа требуется программа Acrobat, и

2. Добавьте скрипт, проверяющий следующие данные:
    - заполнено ли поле имени,
    - введен ли пароль и содержит ли он больше 4-х символов. Используйте для этого свойство length данного поля,
    - совпадают ли значения, введенные в оба поля для паролей,
    - заполнено ли поле электронного адреса и содержит ли оно символ @,
    - заполнено ли поле сообщения и содержит ли оно больше 10 символов,
3. При несоблюдении условий, курсор должен установиться в то поле, где пользователем введено неверное значение.
4. Сохраните документ с именем Ex15.html в рабочей папке.

2.4. Практическая работа №4. Объединение JavaScript и CSS.

Задание 15.

1. Рассмотрите скрипт:

```
<head>
<title>h1</title>
<script language="JavaScript">
function colorchange()
{
head.style.color = "red";
}
</script>
</head>
<body>
<h1 id="head" onmouseover="colorchange()">Добро пожаловать на нашу страницу!</h1>
</body>
</html>
```

2. Допишите скрипт страницы таким образом, чтобы красный цвет исчезал после отвода курсора мыши с заголовка.
3. Сохраните документ с именем Ex15.html в рабочей папке.

Задание 16.

1. Рассмотрите скрипт:

```
<html>
<head>
<title>text decoration</title>
<script language="JavaScript">
function addunderline()
```

```
anil.style.pixelLeft +=50;
setTimeout("moveTxt()", 5000);
}
}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anil" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

2. Измените скрипт страницы:
   - добейтесь плавного передвижения текста;
   - измените направление текста - задайте направление сверху вниз при помощи атрибута pixelTop.
3. Сохраните документ с именем Ex18.html в рабочей папке.

Задание 19.

1. Рассмотрите скрипт:

```
<head>
<title>anima1</title>
<script language="JavaScript">
function moveTxt()
{
if (anim.style.pixelTop <500)
{
anim.style.pixelTop +=2;
anim.style.pixelLeft +=2;
setTimeout("moveTxt()", 50);
}
}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anim" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

2. Измените направление текста. Задайте направление с верхнего правого угла экрана (приблизительно) по диагонали к середине экрана.

Литература

1. Пол Вилтон, Джереми МакПик. JavaScript. Руководство программиста. СПб: Питер, 2009-720 с.
2. Стоян Стефанов. JavaScript. Шаблоны. СПб: Символ-плюс, 2011-272 с.
3. Дунаев В.. HTML, скрипты и стили. СПб: БХВ-Петербург, 2011- 816 с.
4. Дронов В.. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. СПб: БХВ-Петербург, 2011- 416 с.
5. Джон Поллок. JavaScript. Руководство разработчика. СПб: Питер, 2011-544 с.
6. Дэвид Макфарланд. JavaScript. Подробное руководство. Эксмо, 2009-608 с.
7. Климов А. JavaScript на примерах. СПб: БХВ-Петербург, 2009-336 с.
8. Шафер С., HTML, XHTMLи CSS. Библия пользователя. М.: Вильямс, 2010 – 656 с.
9. Лабберс К., Олберс Н., Салим К.. HTML5 для профессионалов: мощные инструменты для разработки современных веб-приложений. М.: Вильямс, 2011 – 272 с.

Интернет-ресурсы

www.w3.org

Область профессиональной деятельности бакалавров и магистров включает:

- сервисно-эксплуационная в сфере современных ИКС;
- расчетно-проектная при создании и поддержке сетевых услуг и сервисов;
- экспериментально-исследовательская;
- организационно-управленческая – в сфере информационного менеджмента ИКС.

Знания выпускников востребованы:

- в технических и программных системах;
- в системах и устройствах звукового вещания, электроакустики, речевой, и мультимедийной информатики;
- в средствах и методах защиты информации;
- в методах проектирования и моделирования сложных систем;
- в вопросах передачи и распределения информации в телекоммуникационных системах и сетях;
- в методах управления телекоммуникационными сетями и системами;
- в вопросах создания программного обеспечения ИКС.

Выпускники кафедры Программных систем обладают компетенциями:

- проектировщика и разработчика структур ИКС;
- специалиста по моделированию процессов сложных систем;
- разработчика алгоритмов решения задач ИКС;
- специалиста по безопасности жизнедеятельности ИКС;
- разработчика сетевых услуг и сервисов в ИКС;
- администратора сетей: UNIX и Windows;
- разработчика клиентских и клиент-серверных приложений;
- разработчика Web – приложений;
- специалиста по информационному менеджменту;
- менеджера проектов планирования развития ИКС.

Трудоустройство выпускников:

1. ОАО «Петербургская телефонная сеть»;
2. АО «ЛЕНГИПРОТРАНС»;
3. Акционерный коммерческий Сберегательный банк Российской Федерации;
4. ОАО «РИВЦ-Пулково»;
5. СПБ ГУП «Петербургский метрополитен»;
6. ООО «СоюзБалтКомплект»;

4. **Oracle** Academy;
5. **Sun Java** Academy и др;
6. **Prometric**;
7. **VUE**.

Мы готовим квалифицированных инженеров в области инфокоммуникационных технологий с новыми знаниями, образом мышления и способностями быстрой адаптации к современным условиям труда.

Редакционно-издательский отдел
Санкт-Петербургского национального
исследовательского университета
информационных технологий, механики
и оптики

197101, Санкт-Петербург, Кронверкский пр., 49