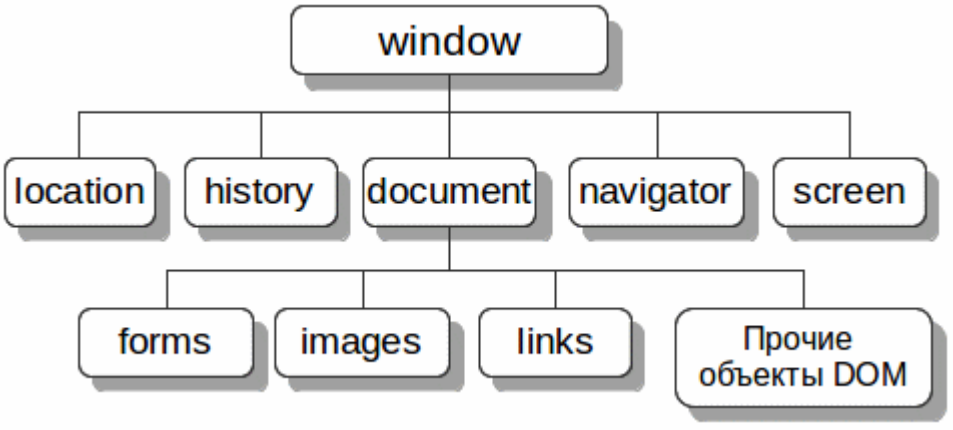
Лабораторная работа №5
Методы в JavaScript

Во время интерпретации HTML-документа браузером создаются объекты JavaScript. Свойства объектов определяются параметрами тегов языка HTML. Структура документа отражается в иерархической структуре объектов, соответствующих HTML-тегам. Родителем всех объектов является объект windows, расположенный на самом верхнем уровне иерархии, он представляет окно браузера и создается при запуске браузера. Для того чтобыоткрыть новое окно в сценарии JavaScript и отобразить в нем новый документ, применяется метод open, для закрытия окна можновоспользоваться методом close. Метод alert объекта windows отображает диалоговое окно с текстом, переданным методу в качестве параметра. Данный метод используется в случаях проверки правильности вводимых данных с помощью формы. Свойства объекта windows относятся ко всему окну, в котором отображается документ.

Подчиненными объектами (или объектами нижнего уровня) являются объекты document, history, location, frame. Свойства объекта history представляют адреса ранее загружаемых HTML-страниц. Свойства объекта location связаны с URL-адресом отображаемого документа, объекта frame - со специальным способом представления данных.

Свойства объекта document определяются содержимым самого документа: шрифт, цвет фона, формы, изображения и т. д. Объект document в зависимости от своего содержимого может иметь объекты, являющиеся для него подчиненными или дочерними. В частности подчиненными для объекта document являются объекты form, image, link, area и др.



Для каждой страницы создается один объект document, некоторые его свойства соответствуют параметрам тега <BODY>: bgColor, fgcolor, linkcolor, alinkcolor, vlinkColor. Методы write и writeln записывают в документ текст, задаваемый параметром.

Если документ содержит изображения, то доступ к объекту, определяющему изображение, можно получить с помощью переменной, указанной в параметре name тега <Img>. Объект image имеет свойство images, которое содержит ссылки на все изображения, расположенные в документе. Ссылки перенумерованы, начиная с нуля. Доступ к первому изображению можно получить с помощью составной конструкции document.images[0], ко второму - document.images[1]. Если на странице пять изображений, то доступ

к последнему изображению можно получить, воспользовавшись ссылкой document.images[4].

Рассмотрим примеры, в которых используются различные свойства объектов.

Для встраивания изображений в HTML-документ служит тег <IMG>, имеющий обязательный параметр src, определяющий URL-адрес файла с изображением. Можно задавать размеры выводимого изображения. Значение

параметра width определяет ширину изображения, значение параметра height - высоту изображения. Значения параметров ширины и высоты могут не совпадать с истинными размерами изображений, тогда при загрузке изображения автоматически выполняется перемасштабирование.

Изображение можно поместить в рамку. Для этого используется параметр border. Значением параметра должно быть число, определяющее толщину рамки в пикселях. По умолчанию рамка вокруг изображения отсутствует, если только изображение не является ссылкой.

Параметр alt определяет альтернативный текст. При наведении курсора мыши на изображение появляется комментарий.

Пример 1. Перестановка изображений

Напишем сценарий, который реализует обмен рисунков в документе. Пусть в документе расположено три изображения, пронумерованных от 1 до 3. В текстовых полях указываются номера рисунков, которые необходимо поменять местами. Требуется, чтобы после нажатия кнопки Обменять изображения переместились на нужные места.

Сначала проверим, правильно ли заданы номера изображений, если это не так, то выдадим сообщение. Переменная z служит для запоминания адреса первого графического изображения. Доступ к изображению с номером r1 производится с помощью конструкции document.images[r1-1]. Для того чтобы на место изображения с номером r1 поместить изображение с номером r2, требуется выполнить оператор присваивания:

document.images[r1-1].src=document.images[r2-1].src

И, наконец, на место изображения с номером r2 помещается изображение, которое ранее было на месте с номером r1, и адрес которого запомнили в переменной z:

document.images[r2-1].src=z

Приведем полностью документ со сценарием (листинг 1).

Листинг 1. Перестановка изображений с помощью сценария

```
<HTML>
<HEAD>
<TITLE>Перестановка изображений</TITLE>
<script>
function chan(obj)
{ var r1=Number(obj.a1.value)
var r2=Number(obj.a2.value)
if ((r1<1)||(r1>3)||(r2<1)||(r2>3))
alert ("Неверно заданы номера рисунков!")
else
{ var z=document.images[r1-1].src
```

```
document.images[r1-1].src=document.images[r2-1].src;
document.images[r2-1].src=z
} }
</script>
</HEAD>
<BODY>
<CENTER>
<H4>Галерея рисунков</H4><br>
<IMG src="p1.gif" width="90" name="pic1">
<IMG src="p2.gif" width="90" name="pic2">
<IMG src="p3.gif" width="90" name="pic3"> <br>
<FORM name=form1>
Рисунки с номерами<br>
<input type="text" name="a1" size=1> и
<input type="text" name="a2" size=1><P>
<input type="button" value="Поменять местами" onClick="chan(form1)">
</FORM>
</CENTER>
</BODY>
</HTML>
```

Пример 2. Простое вертикальное меню

Напишем сценарий, реализующий вертикальное графическое меню.При наведении курсора мыши на пункт меню меняется цветовая палитра, соответствующая выделенному пункту меню.

Каждому пункту меню соответствует два изображения: первое изображение, когда пункт меню не выбран, второе - при выбранном пункте меню, цветовая палитра рисунка изменена. Графические изображения, соответствующие ситуации, когда пункты меню не выбраны, хранятся в файлах с именами pch1.gif, pch2.gif. Соответствующие им графические изображения с измененной палитрой хранятся в файлах с именами wpch1.gif, wpch2.gif.

Функция img имеет два параметра. Первый параметр задает выбор пункта меню, второй параметр - n - определяет номер пункта меню. От этого параметра зависит, какое изображение в документе требуется изменить document.images[n-1].src (вставить на этом месте рисунок "wpch"+n+".gif" или pch"+n+".gif). Имя файла формируется динамически и представляет собой конкатенацию (слияние) строк, одна из составляющих которой - значение второго параметра. Если имена файлов не подчинены общему правилу, то в функции потребуется дополнительный анализ, какой файл подгрузить. Это сделать нетрудно, зная место в документе, из которого произошел вызов функции. Документ со сценарием, реализующий вертикальное графическое меню, представлен в листинге 2.

Листинг 2. Простое вертикальное меню

```
<HTML>
<HEAD>
<TITLE>Простое вертикальное меню</TITLE>
<script
language="JavaScript">
function img(n, action)
{if (action)
{document.images[n-
1].src="wpch"+n+".gif"}else
{document.images[n-1].src="pch"+n+".gif"}
}
</script>
</HEAD>
<BODY background="fon1.jpg">
<H2><FONT color="#0000ff">Содержание</FONT></H2>
<A href="tch1.htm" onmouseover="img(1,1)" onmouseout="img(1,0)">
<IMG src="pch1.gif" alt="форматирование текста" border="0" width="103"
height="35"></A><br>
<A href="tch2.htm" onmouseover="img(2,1)" onmouseout="img(2,0)">
<IMG src="pch2.gif" alt="форматирование текста" border="0" width="103"
height="35"></A><br>
<A href="tch3.htm" onmouseover="img(3,1)" onmouseout="img(3,0)">
<IMG src="pch3.gif" alt="форматирование текста" border="0" width="103"
height="35"></A><br>
</BODY>
</HTML>
```

При попадании курсора мыши в область изображения возникает событие Mouseover, параметр обработки события onMouseOver получает значение img(p1,true).

Задания

1. Проверить примеры лабораторной работы.
2. Написать сценарий выбора из трех изображений одного, которое вставляется ниже этих трех.
3. Написать сценарий картинки с "эффектом приближения", т.е. увеличения размеров как реакция на попадание курсора мыши в поле рисунка (использовать свойства width и height).
4. Написать сценарий графического горизонтального меню с появляющейся стрелкой над пунктом, у которого находится курсор.