Лабораторная работа № 8

Использование java-скриптов в HTML-документе

Цель работы: познакомиться с объектами JavaScript, представляющими браузер (объектная модель браузера — Browser Object Model, BOM) и HTML-документ (объектная модель документа — Document Object Model, DOM), изучить связь между тегами HTML и свойствами каскадных таблиц стилей с одной стороны и объектами Javascript с другой, научиться размещать элементы документа в окне с помощью специальных свойств таблиц стиля CSS, динамически изменять отображение документа в браузере.

Задание

1. Откройте в текстовом редакторе файл **my_css2.html**.
2. Добавьте в документ элемент <script> для вставки команд на Javascript. Вставьте в него строку кода на Javascript, выводящую в окно браузера при открытии документа информацию о последней модификации документа. Проверьте работу скрипта, для запуска скрипта разрешите браузеру исполнение заблокированного содержимого.
3. Добавьте в элемент <script> строку кода, выводящую в строку состояния браузера при открытии документа текст «Начало работы», для этого используйте соответствующее свойство объекта window.
4. В правой части окна документа поместите 4 кнопки с подписями "Window", "Navigator", "Location", "Screen", для выполнения задания используйте средства позиционирования элементов из CSS.
5. Подготовьте обработчик сообщения кнопки "Window", который по нажатию кнопки открывает окно сообщения с текстом, скопированным из строки состояния браузера, после закрытия окна сообщения заменяет содержимое строки состояния (объект window).
6. Подготовьте обработчик для кнопки "Navigator", который выводит информацию об используемом браузере: тип, название, …(объект navigator).
7. Подготовьте обработчик для кнопки "Location", который выводит в окно сообщения имя пользователя и адрес загруженного документа (объект location).
8. Подготовьте обработчик для кнопки "Screen", который выводит информацию о параметрах экрана в окно подтверждения и, если будет нажата кнопка Ок, повторяет вывод в окне браузера (объект screen).
9. Подготовьте скрипты, изменяющие текст в строке состояния при наведении мыши на самый верхний заголовок документа. Для этого:
   - Подготовьте скрипт, изменяющий текст строки состояния на строку «Читай текст!» при наведении мыши на заголовок (сообщение onMouseOver для элемента-заголовка).
   - Подготовить скрипт, изменяющий текст строки состояния на строку «Продолжение работы» при отодвигании мыши (сообщение onMouseOut).
10. Протестируйте работу ваших скриптов.
11. Добавьте скрипты, выполняющие над объектами документа следующие действия:
    - при наведении мыши на первый абзац он изменяет цвет символов и шрифт, скрипт встроить в элемент,
    - при наведении мыши на второй абзац он меняет величину правого поля абзаца - 3 см и отступа первой строки 2,5 см, скрипт оформит в виде функции,
    - при отодвигании мыши от абзацев они восстанавливают свой первоначальный вид.
12. Разработайте для оформления ячеек таблицы два стиля или воспользуйтесь созданными

в предыдущей работе. Разработайте для каждого стиля ячейки альтернативный (инверсный) стиль, в котором меняют значение цвет фона, шрифта или какой-либо другой атрибут. Установите применение альтернативных стилей при наведении мыши на ячейку таблицы и восстановление исходных стилей при отодвигании мыши, для выполнения задания используйте свойство объекта ячейки className.

13. Добавьте три рисунка, рисунки расположите один над другим с небольшим сдвигом, укажите порядок отображения слоев (атрибут zIndex), каждому рисунку присвойте имя (name).
14. Под рисунками расположите три кнопки, с кнопками свяжите обработчики:
    - первый делает невидимым рисунок в среднем слое,
    - второй делает видимым рисунок в среднем слое,
    - третий меняет местами нижний и верхний слои.
15. Проверьте работу скриптов в документе.
16. Результаты покажите преподавателю.

Справочный материал

Создание простейших скриптов

Java-скрипты создаются в элементе <script>, который должен располагаться внутри элемента <head>, или непосредственно в теге элемента html. Если команды javascript находятся непосредственно в элементе <script>, они выполняются при загрузке документа. Если код оформлен в элементе <script> в виде функции или помещен в какой-либо тег, то этот код выполняется, когда происходит какое-либо событие, и является обработчиком этого события.

Пример 1. Простейший скрипт. Выполняется автоматически при загрузке документа. Строчка `document.write("<p><b>Hello World!</b></p>")` выполняет вывод текста в окно браузера с форматированием. Скрипт вставлен в комментарий (`<!-- -->`) — если браузер не умеет обрабатывать скрипты, то он будет игнорировать этот код.

```
<HTML>
<HEAD>
<TITLE>Пример 1</TITLE>
<SCRIPT type="text/javascript">
<!--
document.write("<p><b>Hello World!</b></p>")
-->
</SCRIPT>
</HEAD>
<BODY>
<P>Перевод: Привет, мир!</P>
</BODY>
</HTML>
```

Пример 2. Вызов java-скрипта с помощью кнопки. Функция privet() помещена в элемент <SCRIPT>, выводит текст в стандартное окно сообщений. Функция вызывается при нажатии кнопки (элемент input)— событие `OnClick`. Код, связанный с событием, должен находиться в двойных кавычках.

```
<HTML>
<HEAD>
<TITLE>Пример 2</TITLE>
<SCRIPT type="text/javascript">
function privet()
{
```

```
     alert("Всем привет!");
}
</SCRIPT>
</HEAD>

<BODY>
<FORM name="form1">
Хотите передать привет?
<HR>
Нажмите кнопку
<INPUT type="button" value="Передать привет" OnClick="privet();">
</FORM>
</BODY>
</HTML>
```

Пример 3. Java-скрипт помещен непосредственно в отображаемый элемент html-документа (кнопка). Т.к. код обработчика события помещен в двойные кавычки, текст выводимой строки помещен в одиночные кавычки.

```
<HTML>
<HEAD>
<TITLE>Пример 3</TITLE>
</HEAD>

<BODY>
<FORM name="form1">
Хотите передать привет?
<HR>
Нажмите кнопку
<INPUT type="button" value="Передать привет" OnClick="alert('Всем
привет!');">
</FORM>
</BODY>
</HTML>
```

Создание кнопки

Кнопка — это элемент формы. Для создания формы используется тег form, внутри которого помещаются теги элементов управления (кнопок, надписей, окон ввода, окон со списками):

```
<form name="form_name">       </form>
```

name – атрибут, задающий имя формы.

Для создания элементов управления в форме используется тег input. Кнопка — один из элементов управления — создается следующим образом:

```
<input type="button" value="Текст" onClick="function_name">
```

type – атрибут, задающий тип элемента; значение "button" обозначает кнопку,

value - атрибут, определяющий надпись на кнопке (текст произвольного содержания); тип данных — строка,

onClick – атрибут, определяющий функцию, выполняющуюся при нажатии на кнопку.

Позиционирование элементов

Свойства, задающие положение элемента:

top – y-координата верхней границы объекта,
left – x-координата левой границы объекта,
position – способ определения положения элемента на странице: absolute – положение элемента, задаваемое значениями left и top, определяется относительно левого верхнего угла окна браузера, relative – позиционирование элемента выполняется относительно положения, которое занимал бы элемент по умолчанию.

```
<img src=fish.jpg style="position:absolute; top:450;left:600;">
```

Слои в документе

Создание слоя в документе:

```
<div>   </div>
```

Пример:

```
<div style="position:absolute; top:20; left: 30; width:150;
height:150; background:lime">Это слой с текстом на зеленом фоне,
помещенным в прямоугольную область 150 x 150 пикселей.</div>
```

Управление слоями

z-index – значение этого атрибута определяет номер и соответственно порядок следования слоев. Основной текст находится в слое с номером 0, слой ниже основного имеет отрицательное значение атрибута, слой выше основного – положительное.

```
<div style="position:absolute; top:20; left: 30; width:150;
height:150; z-index=1">Это слой с текстом, помещенным в
прямоугольную область 150 x 150 пикселей (основной текст) на фоне
рисунка.</div>

<div style="position:absolute; top:20; left: 30; z-index=0"><img
src="picture.gif"></div>
```

Отображение объектов

visibility – атрибут стиля, определяющий, виден ли на экране объект: если значение атрибута равно hidden, объект не отображается (скрыт), если атрибут получает значение visible, объект становится видимым. Обращении к объекту осуществляется через свойство объекта name – имя объекта.

Пример:

```
<img src="travel.jpg" name="pic1">

<div>Нажмите на любую кнопку – картинка спрячется!</div>

<form name="f2">

<input type="button" value="Hide" name="b3"
onClick="pic1.style.visibility='hidden'">

</form>
```

Объектная модель браузера

Объект navigator – представляет используемый браузер. С помощью этого объекта можно получить информацию об имени, версии браузера и другую дополнительную информацию.

Свойства объекта navigator

appCodeName	Представляет кодовое имя браузера
appName	Представляет официальное имя браузера
appVersion	Содержит информацию о версии браузера
platform	Хранит информацию о платформе, на которой выполняется браузер
userAgent	Хранит текст заголовка "user-agent"

Пример.

```
<html>
<head>
<title>Объектная модель браузера</title>
<script type="text/javascript">
document.write("Браузер: "+navigator.appName+"<br> Версия:
"+navigator.appVersion);
</script>
</head>
<body>
</body>
</html>
```

Объект window – представляет окно браузера.

Свойства объекта window	
status	Сообщение в строке состояния
Методы объекта window	
open(URL, windowName)	Открывает новое окно браузера
close( )	Закрывает активное окно
alert(message)	Выводит окно предупреждения с сообщением и кнопкой ОК
confirm(message)	Выводит окно подтверждения с двумя кнопками Ok и Cancel, возвращает значение логического типа, которое можно использовать в инструкции if: `if(window.confirm("question")) window.alert("Ответ принят"); else window.alert("Вышлите ответ позже!");`

Примеры:

```
window.status="Мой документ";
window.open("mydoc.htm", "Мой документ");
window.close( );
window.alert("Сообщение");
window.alert("Сообщение"+window.status);
window.confirm("Вы не забыли выключить обогреватель?");
```

Так как объект window представляет верхний уровень объектов, его имя при вызове методов можно опустить: alert("Сообщение");

Объект location — представляет URL-адрес загруженного в браузер документа

href	Полный URL
protocol	Начальный элемент URL до двоеточия – название протокола
hostname	Хост или имя домена или IP-адрес
pathname	Элемент пути URL

Пример:

```
window.alert(location.protocol);
```

Объект screen

width	Ширина экрана
height	Высота экрана
colorDepth	Глубина цвета изображения в пикселях
availWidth	Ширина экрана с учетом служебных элементов окна
availHeight	Высота экрана с учетом служебных элементов окна

Пример:

```
window.alert("Глубина цвета: "+ screen.colorDepth);
```

Объект document

Свойства объекта document	
alinkColor	Цвет активной ссылки (мышь нажата, но еще не отпущена), соответствует <body alink= "color">
bgColor	Цвет фона документа, соответствует <body bgcolor="color">
fgColor	Цвет текста документа, соответствует <body text="color">
linkColor	Цвет непосещенной ссылки, соответствует <body link= "color">
vlinkColor	Цвет посещенной ссылки, соответствует <body vlink= "color">

title	Название документа, определенное в тэге <title>
body	ссылка на элементы, включенные в тэг <body>
lastModified	Дата последнего изменения документа
Методы объекта document	
write(text)	Вывод текста в окно браузера без перевода строки
writeln(text)	Вывод текста в окно браузера с переводом строки
close( )	Закрытие документа

Пример:

```
document.write(document.title);
```

События, связанные с действиями пользователя, и их обработка

onClick	Выполняется при нажатии на кнопку, используется в тэге input.
onMouseOver	Происходит, когда курсор мыши располагается над данным объектом.
onMouseOut	Происходит при отодвигании курсора мыши за пределы объекта.
onLoad	Выполняется, когда завершается загрузка документа в окно браузера или фреймовой структуры, используется в тэгах body или frameset.

Пример :

```
<body onLoad= "document.vlinkColor='green';alert('Зеленые
ссылки!');">
```

Связь между JavaScript, и тегами HTML

Каждый элемент HTML-документа является объектом в Javascript. Обратиться к объекту можно по имени (атрибут name) или через идентификатор (атрибут id). Пример с использованием атрибута name приведен в разделе «Отображение объектов». Также текущий объект можно определить с помощью ключевого слова this:

`<p align="center" onMouseOver="this.innerText='Новый текст!'"><b>Мой абзац</b></p>` - замена текста абзаца при наведении на него мыши.

`<p align="center" onMouseOver="this.style.color='red'"><b>Мой абзац с красными буквами</b></p>` - изменение цвета текста абзаца при наведении на него мыши.

Связь между JavaScript и CSS

Каждому атрибуту таблицы стилей поставлено в соответствие свойство объекта-элемента документа HTML. Сама таблица стилей представляет собой объект style.

Свойства Javascript	CSS	Описание

color	color	Цвет символа
fontSize	font-size	размер шрифта
fontFamily	font-family	семейство шрифтов (название шрифта — Arial и т.д.
fontStyle	font-style	начертание символов: нормальное, курсив
fontWeight	font-weight	жирность шрифта (100, 200, . . 900)
marginLeft	margin-left	ширина левого поля элемента
marginRight	margin-right	ширина правого поля элемента
bgColor	background-color, background	цвет фона
textIndent	text-indent	абзацный отступ (красная строка)

Пример обращения к свойству элемента:

```
<p onMouseOver="this.style.color='red';">Мой абзац с красными буквами</p>
```

Также существует свойство в Javascript — className — для динамического установления правила таблицы стилей, используемого для отображения элемента.

```
<head>
<style type="text/css">
.mystyle {color: red; font-style:"courier"}
.defstyle {color: blue; font-style: "arial"}
</style>
</head>
<body>
…..
<p class="defstyle" onMouseOver="this.className='mystyle'">Абзац</p>
….
</body>
```

При наведении мыши на абзац изменяется стиль абзаца – свойству className присваивается другое значение.

```
<p style="color:'blue'" onMouseOver="this.style.color='magenta'">
```

Для абзаца с внедренной таблицей стиля при наведении мыши изменяется цвет символов.