**Практическая работа 13**

**Строка состояния и таймеры**

**Цель:** Изучить приемы создания строк состояния и таймеров

**Задания:**

1. Ознакомиться с теоретическими сведениями.
2. Создайте документ, содержащий две кнопки, которые можно использовать, чтобы записывать некий текст в строку состояния и, соответственно, затем его стирать.
3. Создайте документ с использованием процедур onMouseOver и onMouseOut.
4. Создайте документ, демонстрирующий кнопку, которая открывает выпадающее окно не сразу, а по истечении 3 секунд.
5. Создайте документ с бегущей строкой в строке состояния.

**Необходимые приборы:** ПК, текстовый редактор Блокнот, браузер

**Методические рекомендации к выполнению лабораторной работы:**

**Методические рекомендации к выполнению задания 1**

**Строка состояния**

Составленные программы на JavaScript могут выполнять запись в строку состояния - прямоугольник в нижней части окна браузера. Все, что необходимо для этого сделать - всего лишь записать нужную строку в *window.status*.

Механизм вывода текста в строку состояния удобно использовать при работе со ссылками. Вместо того, чтобы выводить на экран URL данной ссылки, можно просто на словах объяснять, о чем будет говориться на следующей странице.

**Таймеры**

С помощью функции Timeout (или таймера) можно запрограммировать компьютер на выполнение некоторых команд по истечении некоторого времени.

**Прокрутка**

Можно запрограммировать прокрутку в основной линейке. Рассмотрим также и всевозможные усовершенствования этой линейки. Создать бегущую строку довольно просто. Сперва мы должны записать в строку состояния некий текст. Затем по истечении короткого интервала времени мы должны записать туда тот же самый текст, но при этом немного переместив его влево. Если мы это сделаем несколько раз, то у пользователя создастся впечатление, что он имеет дело с бегущей строкой.

Однако при этом нужно помнить еще и о том, что обязаны каждый раз вычислять, какую часть текста следует показывать в строке состояния (как правило, объем текстового материала превышает размер строки состояния).

Скроллинг используется в Интернет довольно широко. Возможен вариант, когда одна часть текста приходит слева, а другая - справа. И когда они встречаются посередине, то в течение некоторых секунд текст остается неизменным.

**Методические рекомендации к выполнению задания 2**

1 Создайте документ:

<html>

<head>

<script language="JavaScript">

<!-- hide

function statbar(txt) {

window.status = txt;}

// -->

</script>

</head>

<body>

<form>

<input type="button" name="look" value="Писать!"

onClick="statbar('Привет! Это окно состояния!');">

<input type="button" name="erase" value="Стереть!"

onClick="statbar('');">

</form>

</body>

</html>

В этом примере создаются две кнопки, которые можно использовать, чтобы записывать некий текст в строку состояния и, соответственно, затем его стирать.

Начало формы

Конец формы

Обе эти кнопки вызывают функцию statbar(). Вызов от клавиши *Писать!* выглядит следующим образом: statbar('Привет! Это окно состояния!');

В скобках написана строка: *'Привет! Это окно состояния!'*. Это как раз и будет текст, передаваемый функции statbar(). В свою очередь, можно видеть, что функция statbar() определена следующим образом:

function statbar(txt) {

window.status = txt;}

В заголовке функции в скобках мы поместили слово *txt*. Это означает, что строка, которую мы передали этой функции, помещается в переменную *txt*.   
Передача функциям переменных - прием, часто применяемый для придания функциям большей гибкости. Вы можете передать функции несколько таких аргументов - необходимо лишь отделить их друг от друга запятыми.   
Строка *txt* заносится в строку состояния посредством команды window.status = txt.

Соответственно, удаление текста из строки состояния выполняется как запись в *window.status* пустой строки.

**Методические рекомендации к выполнению задания 3**

1. Создайте документ dontclck.htm с произвольным содержанием.
2. Создайте второй документ:

<ahref="dontclck.htm"

onMouseOver="window.status='Don't click me!'; return true;"

onMouseOut="window.status='';">ссылка</a>

3 Сохраните документ под названием Main.htm.

4 Просмотрите результат, запустив в окне браузера документ Main.htm.

В этом примере достаточно лишь поместить указатель вашей мыши над ссылкой и в строке состояния появится соответствующий текст.

Здесь используются процедуры *onMouseOver* и *onMouseOut*, чтобы отслеживать моменты, когда указатель мыши проходит над данной ссылкой.   
В onMouseOver нужно возвращать результат *true* для того, чтобы браузер не выполнял свой собственный код обработки события MouseOver. Как правило, в строке состояния браузер показывает URL соответствующей ссылки. Если же мы не возвратим значение *true*, то сразу же после того, как наш код был выполнен, браузер перепишет строку состояния на свой лад - то есть наш текст будет тут же затерт и читатель не сможет его увидеть.

В некоторых случаях нужно печатать символы кавычек. Например, нужно напечатать текст *Don't click me* - однако поскольку нужно передать эту строку в процедуру обработки события onMouseOver, то для этого используются одинарные кавычки. Между тем, как слово *Don't* тоже содержит символ одинарной кавычки. И в результате если написать *'Don't ...'*, браузер запутается в этих символах. Чтобы разрешить эту проблему, достаточно лишь поставить обратный слэш \ перед символом кавычки - это означает, что данный символ предназначен именно для печати. (То же самое можно делать и с двойными кавычками).

**Методические рекомендации к выполнению задания 4**

1 Создайтедокумент:

<script language="JavaScript">

<!-- hide

function timer() {

setTimeout("alert('Времяистекло!')", 3000);}

// -->

</script>

...

<form>

<input type="button" value="Timer" onClick="timer()">

</form>

В следующем скрипте демонстрируется кнопка, которая открывает выпадающее окно не сразу, а по истечении 3 секунд. Начало формы

Конец формы

Здесь *setTimeout()* - это метод объекта window. Он устанавливает интервал времени. Первый аргумент при вызове - это код JavaScript, который следует выполнить по истечении указанного времени. В нашем случае это вызов - *"alert('Время истекло!')"*. Обратите пожалуйста внимание, что код на JavaScript должен быть заключен в кавычки.

Во втором аргументе компьютеру сообщается, когда этот код следует выполнять. При этом время нужно указывать в миллисекундах (3000 миллисекунд = 3 секундам).

**Методические рекомендации к выполнению задания 5**

1. Введите скрипт:

<html>

<head>

<scriptlanguage="JavaScript">

<!-- hide

// define the text of the scroller

var scrtxt = "Это JavaScript! " +

"Это JavaScript! " +

"Это JavaScript!";

var len = scrtxt.length;

var width = 100;

var pos = -(width + 2);

function scroll() {

// напечатать заданный текст справа и установить таймер

// перейти на исходную позицию для следующего шага

pos++;

// вычленить видимую часть текста

var scroller = "";

if (pos == len) {

pos = -(width + 2);

}

// если текст еще не дошел до левой границы, то мы должны

// добавить перед ним несколько пробелов. В противном случае мы должны

// вырезать начало текста (ту часть, что уже ушла за левую границу

if (pos < 0) {

for (var i = 1; i <= Math.abs(pos); i++) {

scroller = scroller + " ";}

scroller = scroller + scrtxt.substring(0, width - i + 1); }

else {

scroller = scroller + scrtxt.substring(pos, width + pos); }

// разместить текст в строке состо\яни\я

window.status = scroller;

// вызвать эту функцию вновь через 100 миллисекунд

setTimeout("scroll()", 100);}

// -->

</script>

</head>

<bodyonLoad="scroll()">

Это пример прокрутки в строке состояния средствами JavaScript.

</body>

</html>

Большая часть функции scroll() нужна для вычленения той части текста, которая будет показана пользователю. Чтобы запустить процесс прокрутки, используется процедура обработки события onLoad, описанная в тэге <body>. То есть функция *scroll()* будет вызвана сразу же после загрузки HTML-страницы.

Через посредство процедуры onLoad вызывается функция scroll(). Первым делом в функции scroll() устанавливается таймер. Этим гарантируется, что функция scroll() будет повторно вызвана через 100 миллисекунд. При этом текст будет перемещен еще на один шаг и запущен другой таймер. Так будет продолжаться без конца.

**Вопросы для самоконтроля:**

1. Для чего предназначена строка состояния?
2. Можно ли осуществлять управление строкой состояния?
3. Каким образом осуществляется вывод текста в строке состояния?
4. При помощи какой функции можно запрограммировать компьютер на выполнение некоторых команд по истечении некоторого времени?
5. Можно ли запрограммировать прокрутку в основной линейке?