Практическая работа №7
Иерархия объектов. Открытие новых окон.

Цель работы: изучение основных методов и свойств объекта document и особенностей работы с ними

Задачи:
1. Изучить теоретический материал
2. Выполнить практические задания
3. Ответить на контрольные вопросы.
4. Оформить отчет.

Теоретический материал

Все команды начинаются с наивысшего объекта, window (окно браузера), и идут по нисходящей. Окна и рамки (frames) принадлежат объекту window. Их не надо перечислять, если только их не больше одного. Top, self, parent и frames — "встроенные" имена для окон. Вот несколько примеров.

document.pic.src = "pic1.gif" - window не надо ставить в самом начале. Предпо-лагается, что это все и так внутри окна. Команда document.pic.src указывает на рисунок, которому дано имя pic. Документ — это страница, pic — имя элемента, a SRC — источник элемента, "pic1.gif".

document.write(location.href) - write() — это метод объекта document. Location.href показывает полный URL окна. Обратите внимание, что location и document находятся на одном уровне. Это значит, что вы получаете адрес документа того же уровня.

Создание функции

Создавая функцию, вы делаете почти то же самое, только имя присваивается целой серии команд. Множество команд JavaScript вы комбинируете в одну.

Наберите скрипт и посмотрите в браузере результат:

```
<SCRIPT LANGUAGE="javascript">
<!-- Скрыть от браузеров, не читающих Javascript
function dateinbar()

{

var d = new Date();
var y = d.getFullYear();
var da = d.getDate();
var m = d.getMonth() + 1;
var t = da + '/' + m + '/' + y;
defaultStatus = "Вы прибыли на страницу " + t + ".";
}
// не скрывать -->
</SCRIPT>
```

...и команда onLoad в <BODY>:

```
<BODY BGCOLOR="ваш_цвет" onLoad="dateinbar()">
```

Эффект скрипта можно увидеть в строке состояния.

Разбор скрипта

<!-- Это текст комментария, не видимый на странице -->

Если браузер не читает JavaScript, он воспринимает его как текст, который надо напечатать на странице. Но если пользоваться этими командами, тогда браузер успешно проигнорирует незнакомый текст и покажет страницу.

Соблюдайте несколько правил:

• Команды комментария должны находиться между <SCRIPT> и </SCRIPT>.

• Текст после команды <!-- должен находиться на одной строке.

• Перед заключительной командой --> должна стоять двойная дробь //.

Сначала первая часть скрипта устанавливает функцию. Потом команда в строке <BODY> ее запускает. Разберем функцию.

Вы пишете "function" и даете ей любое имя. Оно может быть какой угодно дли-ны, если в нем нет пробелов и это слово уже не участвует в JavaScript.

Внимание, после имени функции стоят круглые скобки, как и после метода.

Команды, из которых состоит функция, должны быть заключены в фигурные скобки.

Команда "onLoad="

Эта команда почти всегда располагается в строке <BODY> документа HTML. И почти всегда за ней следует функция, но это необязательно.

Расположение элементов

Это имеет не последнее значение. Вы знаете, что onLoad идет в строку BODY. Скрипт с функцией должен находиться между командами <HEAD> и </HEAD>. Хотя на самом деле его можно поместить где угодно, но если вы расположите его после команды onLoad, он заработает только после того, как загрузится вся страница. Поместив скрипт перед командой onLoad, вы помещаете его в память компьютера, и когда onLoad вызовет его, он будет готов к работе.

Команды onUnload и onMouseOut

Это два последних обработчика событий, которые вы должны иметь в своем арсенале: onMouseOut и onUnload (обратите внимание на заглавные буквы). Обе они начинают действовать после того, как вы что-то сделали. onMouseOver вызывает некое событие, если навести мышь, к примеру, на ссылку. В противоположность ей onMouseOut начинает действовать, если кур-сор увести со ссылки. Команда onLoad запускает скрипт, когда страница за-гружается. Команда onUnload действует, когда пользователь уходит со стра-ницы.

Скрипт С мышью:

```
<A HREF="les.htm" onMouseOver="window.status='Эй! Уходи от меня!';
return true"onMouseOut="window.status='Так-то лучше, спасибо'; return true">
Наведите курсор на эту ссылку и уведите обратно</A>
При уходе со страницы: <BODY onUnload="alert('Уже уходите?')">
```

Результат

Создайте файл, поместите курсор на ссылку и обратно несколько раз, глядя на строку состояния. Это первый эффект. Нажмите на ссылку, чтобы увидеть второй.

Разбор скрипта

Эффекты с мышью создаются с помощью команд onMouseOver и onMouseOut. Обратите внимание, что между ними ощутимая разница. Вам не надо, чтобы эти события происходили одновременно. Надо писать их как две абсолютно разные команды, каждая из которых содержит свою команду return true. Чтобы получить такой эффект при уходе со страницы, добавляем команду onUnload="alert('Уже уходите?')" в строку BODY. Обратите внимание на двой-ные и одинарные кавычки. Внутри двойных — одинарные. Вторая пара двой-ных кавычек означает для браузера конец команды.

• Создайте страницу с гипертекстовой ссылкой.

• Когда курсор находит на ссылку, в строке состояния появляются слова: «Привет, пользователь название браузера!».

• Когда курсор уходит со ссылки, в строке состояния должен появляться текст: «Не скучаете у нас на URL страницы?»

• Если щелкнуть по ссылке, должно всплыть окно со словами: «Уже уходите? Сейчас всего только текущее время»;

• Время должно определяться через функцию.

Вот скрипт:

```
<head>
<script language=”javascript”>
<!—
var bname = navigator.appName;
```

```
var page = document.location;
function time()
{
var d = new Date();
var h = d.getHours();
var m = d.getMinutes();
var s = d.getSeconds();
var t = h + ':' + m + ':' + s + '';
alert("Куда ж так быстро? Всего только " + t + "!");
}

//-->
</script>
</head>
<body bgcolor="#D6E4E7" onUnload="time()">
<a href="/assig10.htm" onMouseOver="window.status=('Привет, пользователь ' + bname + '!')" return
true
onMouseOut="window.status=('Не скучаете у нас на ' + page + '?')" return true>Пойдем, погуляем?<
/a>
```

Функция между командами <head> задает время. К этой функции обращается команда onUnload в строке <body>. Переменная времени помещается внутри команды alert. Не запутайтесь с двойными и одинарными кавычками. Каждая из команд window.status помещена в двойные кавычки.

Новые окна

Сначала вы узнаете, как через команды Javascript открыть новый документ HTML в другом окне. А также, как с помощью функции поместить две разные страницы в одном документе.

Наберите скрипт и посмотрите в браузере результат: (когда открываем стра-ницу, всплывает второе окно с двумя ссылками)

```
<SCRIPT LANGUAGE="javascript">
window.open('les1.htm', 'joe', config='height=300,width=300')
self.name="main window"
</SCRIPT>
```

Когда вы имеете дело с функцией, скрипт помещается между командами <HEAD>. Если вы собираетесь открывать новое окно, ставьте скрипт ближе к концу документа. Пусть он идет в последнюю очередь. Причина простая: сна-чала загрузится страница, а потом всплывет окошко. Если команда стоит в начале, то окошко всплывет прежде, чем пользователь увидит вашу страницу. Скорее всего он закроет новое окно, не успев им воспользоваться.

window.open: Нельзя сказать яснее, чем это: window (окно) — объект, а open (открыть) — метод, который на него воздействует.

Команды конфигурации: Они сообщают, что новое окно будет размером 300 на 300 пикселей.

Обратите внимание, что команды height (высота) и width (ширина) разделены только запятой без пробелов, а значения поставлены в одинарные кавычки, так как эти два элемента являются подкомандами config. Пробел для браузера означает конец команды. Есть множество подкоманд для команды config. Про высоту и ширину вы уже знаете, они определяются в пикселях. Остальные подкоманды употребляются со словами «yes» или «no» в зависимости от того, нужны ли в новом окне эти элементы. (Можно ставить «1» вместо «да» и «0» вместо «нет».) Помните, никаких пробелов между подкомандами и одинарные кавычки.

- toolbar= отвечает за наличие панели инструментов с кнопками НАЗАД, ВПЕ-РЕД, СТОП и т.д.
- menubar= отвечает за наличие строки меню с элементами ФАЙЛ, ПРАВКА, ВИД и т.д.
- scrollbars= отвечает за наличие полосы прокрутки.

• resizable= указывает, сможет ли пользователь изменить размер окна по сво-ему желанию.
• location= отвечает за наличие адресной строки, где виден URL страницы.
• status= отвечает за наличие строки состояния.

От строки с заголовком вы не избавитесь, хотите вы этого или нет.
Тэги в новом окне
Всплывающее новое окно — это не просто рамка для страницы.
Вот как это получается:
<A HREF="http://www.htmlgoodies.com" TARGET="main window"></A>

У большого окна есть имя, «main window» (главное). Вот почему назвали его главным на протяжении всего урока. В скрипте это обозначено строкой self.name="main window". Добавляем в ссылку HREF TARGET="--" (цель) и ука-зание на main window.

А если надо, чтобы страница загружалась в маленьком окошке, надо написать «joe» после команды target.

С помощью многократных команд window.open можно вызывать многократные окна. Только следите за тем, чтобы у каждого нового окна было свое имя. Можете связывать окна ссылками при условии, что правильно указываете имена окон в команде target.

Закрыть окно: Вторая ссылка нового окна закрыла его.
<A HREF="" onClick="self.close">Щелкните, чтобы закрыть</A>

Это обычная ссылка HREF, которая никуда не ведет. Команда onClick="self.close" закрывает окно и никуда не ведет. self (само, себя) — это свойство может относиться к любому объекту. В нашем случае это свойство окна. Команда close (закрыть) закрывает окно.

Допустим, вы хотите открыть окно по команде, а не когда пользователь захо-дит на страницу:

<A HREF="les11.htm" onClick="window.open('opened.html', 'joe', con-fig='height=300,width=300')">Щелкните, чтобы открыть 'joe'</A>

Посмотрите в браузере, что получилось.

Это ссылка HREF, которая направлена на самое себя. Команда onClick делает работу, а параметры содержатся в скобках().

Создадим функцию, которая откроет новое окно, причем и новое окно, и все его содержимое будет вписано в тот же документ HTML. То есть в букваль-ном смысле слова мы вложим две страницы в одну.

Наберите скрипт и посмотрите результат в браузере:

```
<SCRIPT LANGUAGE="JavaScript">
function openindex()
{
var OpenWindow=window.open("", "newwin", "height=300,width=300");
OpenWindow.document.write("<HTML>")
OpenWindow.document.write("<TITLE>Новое окно</TITLE>")
OpenWindow.document.write("<BODY BGCOLOR='white'>")
OpenWindow.document.write("<CENTER>")
OpenWindow.document.write("<font size=+1>Новое окно</font><P>")
OpenWindow.document.write("<a href= 'www.rumart.net' target='main window'> Этассылка<BR>откроется в главном окне</a><p>")
OpenWindow.document.write("<P><HR><P>")
OpenWindow.document.write("<a href=" onClick='self.close()'> Эта закроет окно</a><p>")
OpenWindow.document.write("</CENTER>")
OpenWindow.document.write("</HTML>")
self.name="main window"
}
</SCRIPT>
```

В строке BODY: onLoad="openindex()"

Помните, текст в скобках должен находиться на одной строке.

Метод confirm (введение в if и else)

Команда confirm (подтвердить) действует очень похоже на alert, за исклю-чением того, что добавляет кнопку «Отмена» в диалоговое окно. И то, и другое — методы. Одна команда сама по себе многого не дает. Нет никакой разницы, что вы выбираете — «ОК» или «ОТМЕНА». Но стоит добавить функции IF (если) и ELSE (иначе), и готовы отличные эффекты.

Наберите скрипты и посмотрите результат в браузере:

```
<SCRIPT LANGUAGE="javascript">
confirm("Уверены, что хотите войти?")
</SCRIPT>
```

Второй скрипт:

```
<SCRIPT LANGUAGE="javascript">
if (confirm("Уверены, что хотите посмотреть?") )
{
parent.location='http://www.rumart.net/book.htm';
alert("Счастливого пути");
}
Else
{
alert("Тогда оставайтесь");
}
</SCRIPT>
```

Разбор скрипта

В нашем случае confirm предлагает альтернативу: «ОК» и «Отмена». Можно сказать, Да и Нет. Обратите внимание на скобки. После команды IF всегда идут скобки, но, как известно, команды confirm тоже требует скобок. Следо-вательно, берем две пары скобок, одна внутри другой. Сразу же после этого идут команды, выполняемые при каждом варианте ответа. Обратите внимание на фигурные скобки{}. Ведь в действительности это функции. Первая из них показывает, что должно произойти, если пользователь выберет ОК (или Да).

parent.location означает ссылку. Дальше идет обыкновенная команда alert. Если выбрали отмену, то следующий текст...

```
else
{
alert("Тогда оставайтесь");
}
```

...означает: если нет, тогда вызвать окно и не менять страницу.

Все это вместе и дает пользователю возможность выбора: входить или не входить.

Ваше задание

Преобразуйте скрипт, о котором говорили выше, в функцию. Сделайте так, чтобы при отмене, кроме окна, еще появлялась какая-нибудь надпись в строке состояния.

Скрипт:

Между командами HEAD:

```
<SCRIPT LANGUAGE="javascript">
function go();
{
if (confirm("Хотите посмотреть?") )
{
parent.location='index.htm';
alert("Счастливого пути");
}
Else
{
alert("Ладно уж, оставайтесь");
alert("Что сделано, то сделано");
}}
</SCRIPT>
```

...и добавка в строку <BODY>: <BODY BGCOLOR="ваш цвет" onLoad="go()">

• Создается имя функции, скрипт копируется и ставится в фигурные скобки;

• К else добавляется defaultStatus='Что сделано, то сделано', которая поме-щает текст в строку состояния;

• Функция запускается командой onLoad в строке BODY;

• Если хотите, чтобы ссылка открылась в новом окне, то надо поменять лишь два слова. Вместо parent.location поставить window.open.

Математические переменные

Наберите скрипт:

```
<BODY>
<SCRIPT LANGUAGE="javascript">
var numsums = 10 + 2
alert("10 + 2 равно " + numsums)
var x = 10
alert("десять — это " + x)
var y = x * 2
alert("10 X 2 = " + y)
var z = "Привет " + "Пока"
alert(z)
</SCRIPT>
</BODY>
```

Посмотрите результат в браузере:

• Переменные начинаются с VAR, следом идет имя, знак = и значение пере-менной.

• Имя переменной может состоять из любого количества букв.

• Имена переменных различают регистр! То есть 'X' и 'x' — это две разные переменные.

• Значение текстовой переменной ставится в кавычки. Числовые переменные не ставятся в кавычки, иначе скрипт поймет их как текст с числовым значением 0!

• Сложение, вычитание, умножение и деление выражаются знаками: +, -, *, и / соответственно.

• Знак плюс (+) выполняет две задачи: складывает числа или печатает вместе два фрагмента текста.

• Во всех языках программирования есть зарезервированные слова, например, названия команд. Этими словами называть переменные нельзя.

• Если необходимо, вместо пробела ставьте знак _ user_name.

Перепишите скрипт в виде функции. Пусть функция запускается командой onLoad.

```
<html>
<head>
<script language="javascript">
<!--
function vars() {
var numsums = 10 + 2;
alert("10 + 2 равно " + numsums);
var x = 10;
alert("десять - это " + x);
var y = x * 2;
alert("10 X 2 = " + y);
var z = "Привет " + "Пока";
alert(z);
}
//-->
</script>
</head>
<body onLoad="vars()">
</body>
</html>
```

Практические задания

Создать страницу со ссылками на выполненные задания данной практической работы.

1. Создайте функцию, которая вызовет два запроса (prompt). (один следует за другим с новой строки.) Первый попросит пользователя ввести свое имя, второй — отчество. Затем та же функция должна вызвать окно предупреждения (alert) с текстом: Привет, имя отчество, добро пожаловать на адрес страницы, страницу!
2. Наберите скрипт, который откроет новое окно со всеми характеристиками. Пусть оно будет размером 250 на 300 пикселей и с двумя ссылками: • Одна откроет новую страницу в главном окне, • Вторая откроет новую страницу в том же окне, •Страница, которая откроется в том же окне, должна содержать ссылку, закрывающую окно.
3. Напишите функцию, которая открыла бы окно с зеленым фоном и приветствием: «Привет, имя пользователя!» Имя пользователя можно узнать с помощью запроса. Допишите еще ссылку, которая закроет окно.
4. Создать веб-страницу, которая бы возводила заданное число в заданную степень. Исходные данные запрашивать через окно ввода. Результат выводить в отдельном окне.
5. Создать функцию, которая вычитает два вводимых пользователем в диалоговое окно числа и выводит сообщение с результатом.
6. Создать функцию, возвращающую наибольшее из трех введенных чисел. Результат выводить в отдельном окне.
7. Напишите функцию, которая будет определять четность числа и окрашивать страницу в определенный цвет в зависимости от точности. Результат выводить в отдельном окне.
8. Создать веб-страницу, запрашивающую три числа и определяющую, возможно ли построить треугольник с такими длинами сторон. Результат выдать в виде сообщения.
9. Создать функцию, определяющую образуют ли 4 введенных числа возрастающую или убывающую последовательность. Результат выдавать в виде сообщения.
10. Напишите функцию, вычисляющую факториал введенного числа. Выдать результат в отдельном окне. Цвет окна должен различаться для четных и нечетных чисел.

Контрольные вопросы

1. Как обратиться к окну браузера?
2. Отличие комант onLoad onUnload?
3. Команды работы с мышью.
4. Как открыть существующий документ в новом окне?
5. Как создать «вложенные» страницы?
6. Назначение команды confirm?
7. Как обеспечить закрытие текущего окна с помощью ссылки?

Содержание отчета

1. Титульный лист
2. Цели, задачи работы
3. Текст задания
4. Листинг
5. Результаты работы
6. Ответы на контрольные вопросы