МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ГЛАВНОЕ УПРАВЛЕНИЕ ПО ОБРАЗОВАНИЮ

МОГИЛЁВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

|  |  |
| --- | --- |
| Специальность  Учебная дисциплина | 2-40 01 01  Программные средства  создания Интернет-приложений  УТВЕРЖДАЮ  Заместитель директора  по учебной работе  \_\_\_\_\_\_\_\_\_\_М.М.Федоськова  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**ЛАБОРАТОРНАЯ РАБОТА № 15**

**ОРГАНИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ С ПРОГРАММОЙ ПРОСМОТРА HTML-СТРАНИЦ СРЕДСТВАМИ BOM**

**МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ**

Разработал преподаватель

Денисовец Д.А.

2021

Обсуждено и одобрено

на заседании цикловой комиссии

специальных дисциплин специальности

«Программное обеспечение

информационных технологий»

Протокол № \_\_ от \_\_\_\_\_\_\_\_\_\_\_\_

**1 Цель работы**

1.1 Формирование умений взаимодействовать с программой просмотра HTML-страниц средствами BOM

**2 Методическое и материальное обеспечение**

2.1 Персональный компьютер

2.2 Методические рекомендации по выполнению лабораторной работы

2.3 Текстовый редактор Sublime Text 3 / notepad++

**3 Последовательность выполнения работы**

3.1 Ознакомьтесь с основными теоретическими положениями.

3.2 Выполните индивидуальное задание.

3.3 Ответьте на контрольные вопросы.

3.4 Оформите отчет.

**4 Основные теоретические положения**

Объектная модель документа имеет иерархическую структуру. На вершине этой иерархии находится объект window . Он имеет следующие дочерние объекты:

* navigator;
* screen;
* history;
* location;
* document.

В скриптах, в основном, используется объект document, так как он целиком описывает загружаемый в браузер HTML-документ. Объекты navigator, screen и history используются для справки о свойствах браузера и компьютера пользователя. Объект location удобно использовать для загрузки страниц или фреймов.

С помощью объектной модели браузера (Browser Object Model) можно управлять поведением браузера из JavaScript.

Для каждого открытого окна браузер создает объект **Window.** Все объекты JavaScript, включая переменные и функции, хранятся в объекте **window**. Методы объекта: **open()** открывает новое окно, **close()** закрывает окно, **alert()** выводит окно оповещения.

Window включает перечисленные ниже объекты.

Объект **Navigator** позволяет определить, какой браузер использует пользователь. Следующий пример показывает, как с помощью свойства **userAgent** можно вывести на страницу информацию о браузере пользователя.

document.write('Информация о браузере:'+navigator.userAgent);

С помощью свойства **appVersion** можно узнать версию браузера, свойства **appName** – название браузера, свойства **appCodeName** – кодовое название браузера, свойства **platform** – ОС, которую использует пользователь. Метод **javaEnabled()** объекта **Navigator** дает возможность проверить, включена ли поддержка Java в браузере пользователя или нет. Метод вернет true, если поддержка включена, и false, если нет.

Объект **Screen** содержит информацию об экране пользователя. Свойство **width** определяет ширину экрана пользователя, свойство **height** – высоту. С помощью свойства **colorDepth** можно узнать глубину цвета (измеряется в битах на пиксель).

Объект **History** содержит список URL, которые были посещены в данном окне браузера. С помощью свойства **length** можно узнать количество посещенных URL, хранящихся в списке.

Объект **Location** позволяет узнать информацию о URL текущего документа. Свойство **href** хранит URL текущего документа целиком. С помощью свойства **pathnam** можно узнать путь к загруженному документу. Свойство **host** содержит имя домена загруженного документа. Метода **assign()** позволяет загрузить новый документ в данное окно браузера (изменить текущий URL на желаемый).

**4.1** **Свойства объекта window**

Объект window имеет свойства, описывающие размеры окна, расположенные в окне фреймы, имя окна, содержимое строки состояния окна и другие, представленные в таблице 1:

Таблица 1 - Свойства объекта window

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| defaultStatus | Сообщение, отображаемое в строке состояния окна браузера по умолчанию |
| frames | Массив всех фреймов данного окна |
| length | Количество фреймов в родительском окне |
| name | Имя окна, указанное при его открытии методом open, а также в параметре TARGET оператора <A> или в параметре NAME оператора <FORM> |
| parent | Синоним имени окна. Относится к окну, содержащему набор фреймов |
| self | Синоним имени окна. Относится к текущему окну |
| status | Текущее сообщение, отображаемое в строке состояния окна браузера |
| top | Синоним имени окна. Относится к окну верхнего уровня |
| window | Синоним имени окна. Относится к текущему окну |

Что касается свойства defaultStatus, то оно используется только в браузере Netscape Navigator. Если записать в это свойство произвольное сообщение, оно

будет отображаться в строке состояния Netscape Navigator, когда окно браузера выдвигается на передний план. К сожалению, браузер Microsoft Internet Explorer версий 3.02 и 4.0 игнорирует свойство defaultStatus.

Свойство status, напротив, применяется обоими конкурирующими браузерами. Периодически записывая сообщение в это свойство и каждый раз сдвигая это сообщение на одну позицию, можно добиться эффекта бегущей строки. Применение строки состояния для отображения бегущей строки распространено достаточно широко.

Заметим, однако, что в браузере Microsoft Internet Explorer версии 4.0 строка состояния разделена на несколько областей. Область, содержимое которой соответствует свойству status, расположена слева и имеет относительно небольшие размеры (особенно в режимах видеоадаптера с низким разрешением). Это необходимо учитывать, если предполагается использовать бегущую строку для привлечения внимания пользователя.

Свойства windows и self - синонимы. Их можно применять по своему усмотрению.

Остальные свойства, в частности, свойства frames и length, применяются в том случае, когда в окно загружен документ HTML с фреймами. Анализируя свойство length можно определить количество фреймов в окне, а при помощи свойства frames (которое является массивом) нетрудно получить доступ к окнам этих фреймов.

**4.2 Методы объекта window**

Среди методов, определенных в объекте window, отметим методы, предназначенные для открытия новых окон и закрытия существующих, для отображения на экране простейших диалоговых панелей с сообщениями и методы для установки таймера, представленные в таблице 2:

Таблица 2 - Методы объекта window

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| alert | Отображение диалоговой панели Alert с сообщением и кнопкой OK |
| close | Закрытие окна |
| confirm | Отображение диалоговой панели Confirm с кнопками OK и Cancel |
| open | Открытие окна |
| prompt | Отображение диалоговой панели Prompt с полем ввода |
| setTimeout | Установка таймера |
| clearTimeout | Сброс таймера |

**Метод alert:**

**Приведем формат вызова метода alert:**

alert("Сообщение");

Через единственный параметр методу alert передается сообщение, отображаемое в диалоговой панели.

После вызова этого метода выполнение сценария (точнее говоря, функции сценария, вызвавшей этот метод) задерживается до тех пор, пока пользователь не нажмет кнопку OK, расположенную в диалоговой панели с сообщением.

Заметим, что при вызове метода alert не указан объект, для которого вызывается метод - объект window. Если при вызове метода объект не указан, интерпретатор сценария, встроенный в браузер, предполагает, что метод относится к объекту window.

Тем не менее, можно явно указывать объект window:

window.alert("Сообщение");

Результат будет тем же самым.

**Метод confirm:**

С помощью метода confirm возможно отобразить на экране диалоговую панель с сообщением, однако в этой панели будет две кнопки - OK и Cancel. В зависимости от того, какая кнопка будет нажата, метод возвратит, соответственно, значение true или false.

**Метод confirm обычно используется следующим образом:**

if(confirm("Сообщение"))

{

// Нажата кнопка OK

. . .

}

else

{

// Нажата кнопка Cancel

. . .

}

**Метод prompt:**

Если в сценарии необходимо получить от пользователя одну текстовую строку, для этого можно применить метод prompt. Этот метод отображает на экране диалоговую панель, в которой есть одно текстовое поле ввода и кнопка OK. Когда пользователь нажимает эту кнопку, метод prompt возвращает введенную строку.

**Метод prompt вызывается следующим образом:**

Var szInput="";

szInput=prompt("Сообщение","Строка ввода по умолчанию");

Через первый параметр методу передается сообщение, которое отображается в диалоговой панели над текстовым полем ввода. Второй параметр необязательный. Если он указан, поле ввода инициализируется текстовой строкой, заданной этим параметром.

Заметим, что с помощью метода prompt сценарий может получить от пользователя только одну текстовую строку. В тех случаях, когда необходимо ввести и обработать сразу несколько строк, можно создать в документе HTML форму, имеющую произвольное количеством полей и кнопку завершения ввода с обработкой события. Когда пользователь нажмет эту кнопку, обработчик соответствующего события сможет извлечь из полей формы введенные значения. Подробно работа с формами будет рассмотрена в отдельной главе нашей книги.

**Метод open:**

С помощью метода open сценарий может открыть новое окно браузера и загрузить в это окно документ HTML для просмотра.

**Формат вызова метода open приведен ниже:**

open("Адрес URL", "Имя Окна", "Параметры окна");

Метод возвращает имя нового окна, которое можно использовать для ссылки на свойства и методы окна, а также на свойства и методы объектов, расположенных в этом окне.

Первый параметр метода open задает адрес URL документа HTML, предназначенный для загрузки в новое окно.

Второй параметр определяет имя окна для использования в параметре TARGET оператора <A> или в операторе <FORM>. МОжно указать его как пустую строку вида “”.

Третий, необязательный параметр, задает различные параметры, определяющие внешний вид создаваемого окна браузера. Этот параметр указывается как текстовая строка, где через запятую перечислены значения отдельных параметров, например:

var wndNewWindow;

wndNewWindow=open("hello.html", "",

"toolbar=no,menubar=no,scrollbars=no,width=300,height=100");

Ниже перечислены все возможные параметры окна, представленные в таблице 3:

Таблица 3 - Параметры окна

|  |  |
| --- | --- |
| **Параметр** | **Описание** |
| toolbar | Если параметр имеет значение yes или 1, окно снабжается стандартной инструментальной линейкой. Если же значение этого параметр равно no, то инструментальная линейка будет отсутствовать |
| location | Параметр определяет, будет ли отображаться поле ввода адреса документа |
| directories | Аналогично предыдущему, но управляет отображением кнопок каталогов браузера Netscape Navigator, таких как "What's New" и "What's Cool" |
| status | Отображение строки состояния |
| menubar | Отображение линейки меню |
| scrollbars | Отображение полос просмотра |
| resizable | Если этот параметр указан как yes или 1, пользователь сможет изменять размер вновь созданного окна |
| width | Ширина окна в пикселах |
| height | Высота окна в пикселах |

**Метод close:**

С помощью метода close можно закрыть созданное или основное окно браузера. **Формат вызова этого метода такой:**

wndNewWindow.close()

Заметим, что текущее окно браузера (то есть окно, в которое загружен документ HTML с работающим сценарием) может быть закрыто одним из двух следующих способов:

window.close()

self.close()

Этот метод будет использоваться в сценарии, описанном ниже.

**Метод setTimeout:**

С помощью метода setTimeout можно установить таймер, указав при этом выражение JavaScript и задержку во времени:

idTimer=setTimeout(cmd, timeout);

Метод setTimeout создает и запускает таймер, возвращая его идентификатор. Когда пройдет время, заданное вторым параметром timeout (в миллисекундах), запускается выражение JavaScript, заданное параметром cmd.

Рассмотрим такой фрагмент сценария:

var cmd="NoAccess()";

idTimer=window.setTimeout(cmd, 10000);

Здесь создается таймер с временем задержки 10 секунд. После прошествия этого времени будет вызвана функция с именем NoAccess, которая должна быть определена в сценарии заранее. Эта функция будет выполнена только один раз.

Ниже представлены примеры сценариев, в которых метод setTimeout используется для периодического вызова функции сдвига сообщения в строке состояния браузера (“бегущая” строка) и для ограничения времени ввода пользователем строки пароля.

Еще одно применение метода setTimeout - создание анимационных эффектов в сценарии JavaScript.

**Метод clearTimeout:**

С помощью метода clearTimeout можно остановить таймер, запущенный только что рассмотренным методом setTimeout. В качестве параметра методу clearTimeout необходимо передать идентификатор таймера, полученный от метода setTimeout:

clearTimeout(idTimer);

**4.3 События для объекта window**

С объектом класса window связаны два события - onLoad и onUnload. Первое из них возникает, когда браузер заканчивает загрузку окна или всех окон фреймов, определенных оператором <FRAMESET>, а второе - когда пользователь завершает работу с документом HTML.

В своем сценарии можно предусмотреть обработку этих событий, назначив для каждого из них свою функцию. Функция, которая вызывается при завершении загрузки документа, может выполнять какие-либо инициализирующие действия, создавать дополнительные окна или выводить сообщения. Обработчик события onUnload может освобождать полученные ресурсы или выводить какие-либо дополнительные сообщения.

**В качестве примера рассмотрим следующий фрагмент документа HTML, содержащий сценарий:**

. . .

function Hello()

{

window.alert("Welcome to my home page!")

}

function Bye()

{

window.alert("Bye! Come back again!")

}

. . .

<BODY BGCOLOR=white onLoad="Hello()" onUnload="Bye()">

. . .

Здесь в операторе <BODY> определены обработчики событий onLoad и onUnload. При возникновении первого события будет вызываться функция Hello, а при возникновении второго события - функция Bye. Заметим, что так как документ HTML интерпретируется в направлении сверху вниз, функции Hello и Bye необходимо определить до появления оператора <BODY>. Лучшее место для определения этих функций - заголовок документа HTML.

Если нужно проследить загрузку всех фреймов, можно указать обработчик события onLoad в операторе <FRAMESET>.

1. **Индивидуальное задание**

Напишите программу решения задачи на языке программированияJavaScript в соответствии с заданным вариантом.

**Вариант 1**

1 Создайте кнопку, при нажатии на которую выводится окно подтверждений.

2 Выведите окно подтверждений при загрузке WEB - страницы, если выбрали «OK» - загрузите WEB - страницу, в противном случае прервите загрузку.

**Вариант 2**

1 Создайте кнопку, при нажатии на которую выводится окно приглашения.

2 Загрузите графическое изображение, имя файла получите через окно приглашения.

**Вариант 3**

1 Создайте кнопку, при нажатии на которую выводится окно предупреждения.

2 Выведите окно предупреждения при загрузке WEB - страницы.

**Вариант 4**

1 Создайте кнопку, при нажатии на которую выводится окно подтверждений.

2 Выведите окно подтверждений при закрытии WEB - страницы, если выбрали «OK» - закройте WEB - страницу, в противном случае отмените закрытие WEB - страницы.

**Вариант 5**

1 Создайте кнопку, при нажатии на которую выводится окно предупреждения.

2 Создайте список и выведите окно предупреждения при нажатии на элементы списка.

**Вариант 6**

1 Создайте кнопку, при нажатии на которую выводится окно приглашения.

2 Запросите у пользователя его имя с помощью окна приглашения.

**Вариант 7**

1 Создайте кнопку, при нажатии на которую выводится окно подтверждений.

2 Выведите окно подтверждений при загрузке WEB - страницы, если выбрали «OK» - загрузите WEB - страницу, в противном случае прервите загрузку.

**Вариант 8**

1 Создайте кнопку, при нажатии на которую выводится окно предупреждения.

2 Выведите окно предупреждения при загрузке WEB-страницы.

**6 Содержание отчета (в электронном виде, в рабочей папке учащегося)**

6.1 Тема работы

6.2 Цель работы

6.3 Индивидуальное задание (текст программы и скриншоты выполнения)

6.4 Ответы на контрольные вопросы

**7 Контрольные вопросы**

7.1 Опишите методы объекта window в языке JavaScript.

7.2 Опишите свойства объекта window в языке JavaScript.

7.3 Охарактеризуйте объект window в языке JavaScript.

**Список используемых источников**

1. Закас, Н. JavaScript для профессиональных веб-разработчиков / Н. Закас. - СПб. : Питер, 2015. - 960 с.
2. Флэнаган, Д. JavaScript. Подробное руководство, 6-е издание. – Пер. с англ. / Д. Флэнаган. - Санкт-Петербург: Символ-Плюс, 2012. – 1008 с. : ил.