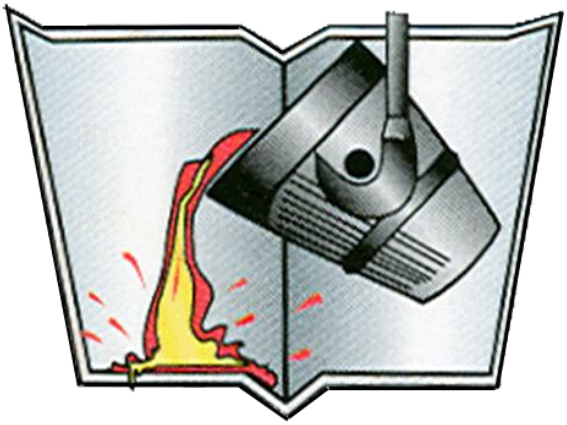
УПРАВЛЕНИЕ ОБРАЗОВАНИЯ И НАУКИ ЛИПЕЦКОЙ ОБЛАСТИ

ГОАПОУ «Липецкий металлургический колледж»



Методические указания по организации и проведению самостоятельной работы студентов по

ПМ 04 Использование пакетов прикладных программ в формировании электронного документооборота предприятия

ПМ 02 Разработка, внедрение и адаптация программного обеспечения отраслевой направленности

на тему:

«Основы создания и управления динамическим контентом: JavaScript»

для специальности (группы специальностей):

09.02.04 Информационные системы (по отраслям)

09.02.05 Прикладная информатика (по отраслям)

Л и п е ц к - 2 0 1 8

Методические указания по организации и проведению самостоятельной работы студентов специальности 09.02.04 Информационные системы (по отраслям) при освоении вида деятельности (ПМ 04) Использование пакетов прикладных программ в формировании электронного документооборота предприятия и специальности 09.02.05 Прикладная информатика (по отраслям) при освоении вида деятельности (ПМ 02) Разработка, внедрение и адаптация программного обеспечения отраслевой направленности.

Разработчики: *Радченко Т. И.,* преподаватель общепрофессиональных дисциплин и профессиональных модулей*;*
Логунова Г.В., преподаватель общепрофессиональных дисциплин и профессиональных модулей.

ОДОБРЕНО Цикловой комиссией информационных систем	УТВЕРЖДАЮ Заместитель директора по учебной работе:
Председатель:	
_____ */Радченко Т. И./*	_____/*Левина Н.М./*

Методические указания по организации и проведению самостоятельной работы студентов предназначены для обучающихся ГОАПОУ «Липецкий металлургический колледж» специальности 09.02.04 Информационные системы (по отраслям) и специальности 09.02.05 Прикладная информатика (по отраслям) для освоения знаний, умений и профессиональных компетенций, получения практического опыта проектирования и выполнения индивидуальных заданий, курсового и дипломного проектирования по специальности.

Методические указания по организации и выполнению самостоятельной работы студентов составлены в соответствии с ФГОС СПО ППССЗ специальности 09.02.04 Информационные системы (по отраслям) и специальности 09.02.05 Прикладная информатика (по отраслям).

Содержание

	Стр.
Введение	4
1 Методические указания по выполнению самостоятельной работы для студентов	6
2 Техника безопасности при выполнении самостоятельной работы на персональном компьютере	7
3 Теоретический материал	8
3.1 Основные понятия	8
3.2 Что такое JavaScript?	8
3.3 Что может JavaScript?	9
4 Практическое применение JavaScript	10
4.1 Встроенный сценарий	10
4. 2 Внешний сценарий	14
5 Теория построения программного кода JavaScript	16
5.1 Синтаксические особенности кода JavaScript	16
5.2 Переменные в JavaScript	17
Задание для самостоятельного выполнения	21
Приложение А. Примеры кода сценария JavaScript	23
Перечень рекомендуемых источников	24

Введение

Методические указания по проведению самостоятельной работы студентами разработаны с целью реализации рабочей программы ПМ 04 Использование пакетов прикладных программ в формировании электронного документооборота предприятия для специальности 09.02.04 Информационные системы (по отраслям) и рабочей программы ПМ 02 Разработка, внедрение и адаптация программного обеспечения отраслевой направленности для специальности 09.02.05 Прикладная информатика (по отраслям) в соответствии с требованиями ФГОС СПО и содержанием ППССЗ.

Согласно требованиям ППССЗ теоретический материал и задания для самостоятельного выполнения, изложенные в методических указаниях для специальности 09.02.04 Информационные системы (по отраслям) направлены на освоение следующих умений и знаний

умений:

– подбирать оптимальную программу для создания электронного документа;

– разрабатывать алгоритм создания и редактирования электронного документа;

– создавать интегрированные (связанные) документы разных форматов данных.

знаний:

– среды и инструментов редактирования и форматирования различных объектов электронного документа;

– правил обмена данными межу электронными документами;

для специальности 09.02.04 Прикладная информатика (по отраслям) направлены на освоение следующих умений и знаний

умений:

– идентифицировать, анализировать и структурировать объекты информационного контента;

– разрабатывать информационный контент с помощью языков разметки;

– разрабатывать программное обеспечение с помощью языков программирования информационного контента;

– размещать информационный контент в локальных и глобальных сетях;

– использовать инструментальные среды поддержки разработки, системы управления контентом;

– создавать анимации в специализированных программных средах;

– работать с мультимедийными инструментальными средствами;

– использовать системы управления контентом для решения поставленных задач;

знаний:

– специализированного программного обеспечения проектирования и разработки информационного контента;

– технологических стандартов проектирования и разработки информационного контента;

– принципов построения информационных ресурсов;

– основ программирования информационного контента на языках высокого уровня;

– компьютерных технологий представления и управления данными;

– принципов создания информационных ресурсов с помощью систем управления контентом.

1 Методические указания по выполнению самостоятельной работы для студентов

1. К выполнению задания рекомендуется приступать после изучения основных понятий и определений Web технологии, освоения теории построения гипертекстового документа, получения практического опыта разметки Web-страницы средствами HTML – кода и форматирования средствами каскадных таблиц стилей (CSS).
2. Перед выполнением заданий необходимо изучить лекционный материал и теорию, изложенную в методических рекомендациях. Кроме описания работы в данном учебном пособии, используйте рекомендованную литературу и интернет - источники.
3. Результаты выполнения самостоятельной работы сохраняются в папку студента - портфолио электронных работ студента.
4. Результаты выполнения самостоятельной работы предоставляются преподавателю для проверки в электронном виде.
5. При подготовке к защите самостоятельной работы, необходимо ответить на предложенные контрольные вопросы.

2 Техника безопасности при выполнении самостоятельной работы на персональном компьютере

Работать надо на расстоянии 60-70 см, соблюдая правильную посадку, не сутулясь, не наклоняясь; пользователям, имеющим очки для постоянного ношения - в очках.

2.1 Во время работы:

- необходимо аккуратно обращаться с проводами;
- запрещается работать с неисправным компьютером;
- нельзя заниматься очисткой компьютера, когда он находится под напряжением;
- нельзя располагать рядом с компьютером жидкости, а также работать с мокрыми руками;
- нельзя в процессе работы на ПК прикасаться к другим металлическим конструкциям (например, батареям);
- не допускается употребление пищи в непосредственной близости от компьютерной техники.

2.2 По окончании работы необходимо:

- выключить компьютер;
- навести порядок на рабочем месте места.

2.3 Находясь в компьютерной лаборатории

Учащийся допускается к работе на ПК в сроки, указанные в расписании занятий или при наличии свободных мест в лаборатории ПЭВМ.

Студент должен приходить на самостоятельное занятие подготовленным, с четко определенной целью (во избежание непроизводительного расходования машинного времени).

Студент обязан соблюдать в лаборатории тишину, чистоту и порядок, бережно относиться к аппаратуре. Запрещено приносить в аудиторию большие сумки, верхнюю одежду, пищевые продукты, напитки и прочие посторонние вещи, оставлять в помещении бумаги, газеты и другие предметы.

Никогда не пытайтесь самостоятельно устранять неисправности в работе аппаратуры.

3 Теоретический материал

3.1 Основные понятия

JavaScript - язык сценариев или скриптов применяется для придания интерактивности страницам гипертекстовых документов (Web -страницам).

JavaScript создан программистом Брендоном Ичем (Brendan Eich) в компании Netscape и представлен в декабре 1995 года под названием LiveScript. Новый продукт переименован в JavaScript, но официальным названием JavaScript является ECMAScript.

ECMAScript разрабатывается и поддерживается Международной организацией ECMA - Европейская ассоциация производителей компьютеров.

JavaScript появился во всех браузерах начиная с 1996 года.

J**avaScript** общедоступен и бесплатен.

3.2 Что такое JavaScript?

- JavaScript создан для добавления интерактивности в HTML страницы;
- JavaScript - это язык сценариев или язык скриптов;
- JavaScript - это облегченный язык программирования;
- JavaScript вставляется прямо в HTML страницы;
- JavaScript является интерпретируемым языком (скрипты исполняются без предварительной компиляции).

Скрипт представляет собой программный код — набор инструкций, который не требует предварительной обработки перед запуском. Код JavaScript интерпретируется движком браузера во время загрузки веб-страницы. Интерпретатор браузера выполняет построчный анализ, обработку и выполнение исходной программы или запроса.

JavaScript — объектно-ориентированный язык с прототипным наследованием, поддерживает несколько встроенных объектов, позволяет создавать или удалять пользовательские объекты. Объекты могут наследовать свойства непосредственно друг от друга, образуя цепочку объект-прототип.

3.3 Что может JavaScript?

– JavaScript дает HTML дизайнерам инструмент программирования с очень простым синтаксисом! Почти каждый может вставить небольшие "куски" кода в HTML страницы;

– JavaScript реагирует на события - JavaScript может быть настроен на выполнение определенных действий, когда что-то происходит, например, когда страница загружается полностью или когда пользователь щелкает мышью на HTML элементе;

– JavaScript может читать и писать HTML элементы;

– JavaScript может использоваться для проверки данных, введенных в поля формы, прежде чем они будут переданы на сервер, что экономит время и ресурсы сервера;

– JavaScript может использоваться для определения браузера посетителя, и - в зависимости от того, какой браузер, - загружать другую страницу, спроектированную специально для этого браузера;

– JavaScript может использоваться для создание cookies - сохранения и загрузки информации с пользовательского компьютера;

4 Практическое применение JavaScript

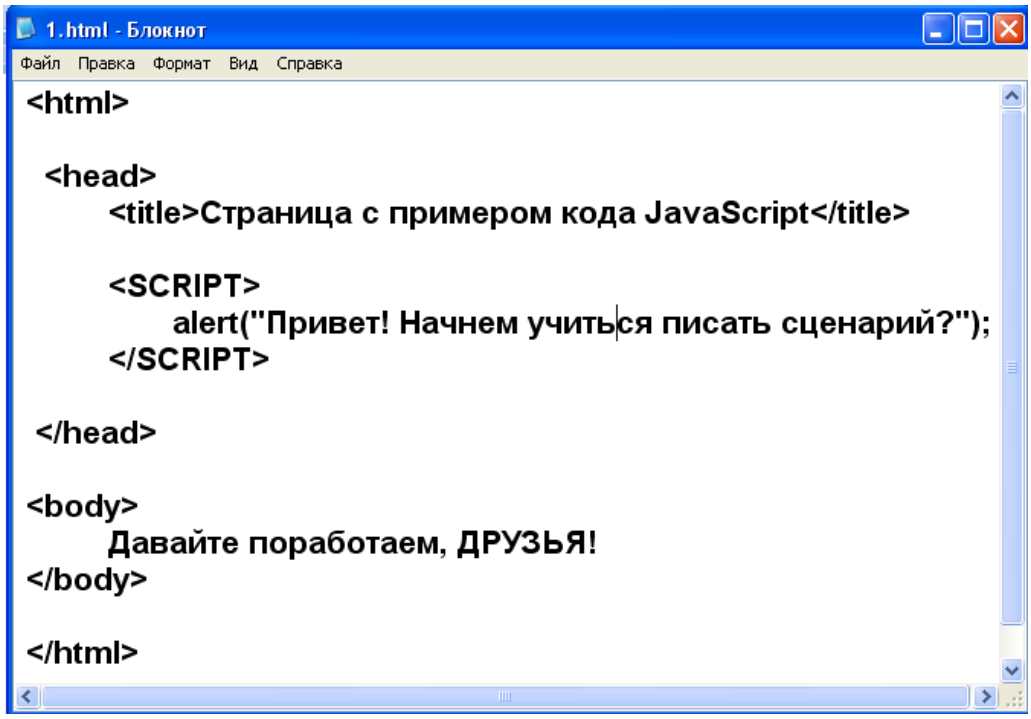
Сценарии JavaScript бывают **встроенные**, когда скрипт прописывается непосредственно в HTML-коде файла и **внешние**, хранящиеся в отдельном файле типа ***.js**.

4.1 Встроенный сценарий

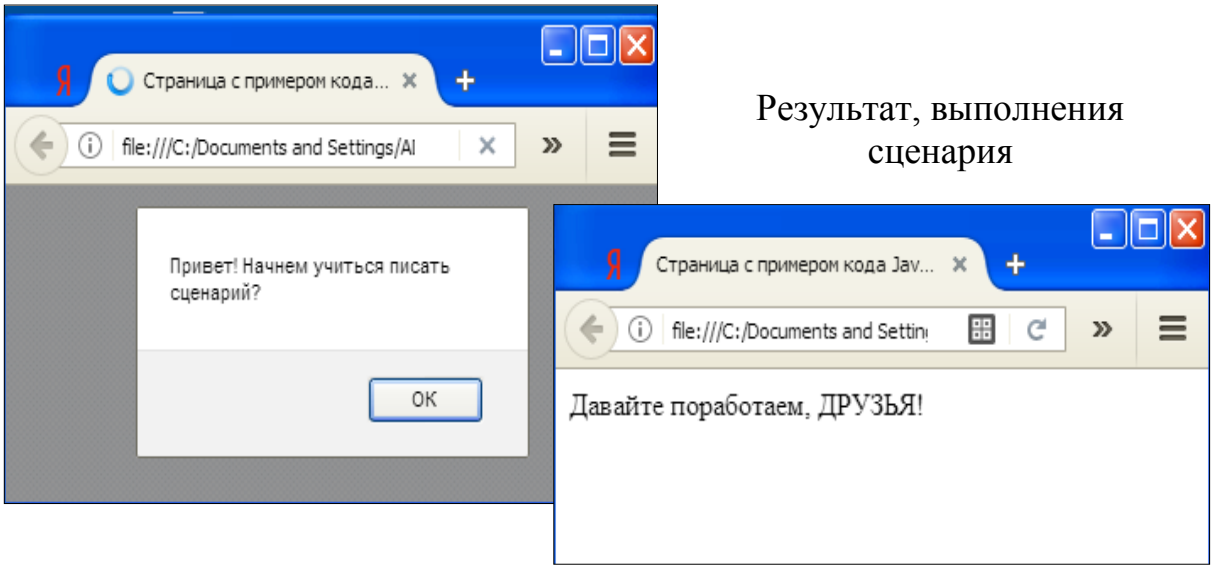
При размещении сценария внутри HTML-кода JavaScript прописывается в теге - контейнере **<script> ... </script>**

```
<script>
        сценарий…
</script>
```

Например, чтобы при открытии страницы появлялось окно диалога с текстом пользователя, откройте блокнот, наберите приведенный ниже код и сохраните файл с расширением ***.html**



При просмотре файла в любом браузере появится модальное окно с надписью «*Привет! Начнем учиться писать сценарий?*» и кнопкой "OK". После клика по "OK" продолжится выполнение страницы и на ней появится текст *"Давайте поработаем, ДРУЗЬЯ!"*.



Результат, выполнения сценария

Для установления диалога с посетителем Web – публикации можно использовать три модальных окна – соответственно в сценарии JavaScript прописывается один из трех методов:

alert();

confirm();

prompt().

Внимание: диалоговое окно, которое появляется в результате выполнения одного из методов alert(), confirm() или prompt() является модальным, оно блокирует доступ пользователя к браузеру до тех пор, пока пользователь не закроет это диалоговое окно.

Метод alert( )

Метод *alert(“текст”)* выводит диалоговое окно с сообщением, которое программист прописал в скобках. Текст сообщения заключается в двойные кавычки. Посетитель не сможет продолжить работу, пока не нажмет на кнопку "OK" в окне диалога.

Этот сценарий позволяет акцентировать внимание посетителя публикации «на важном…».

Метод confirm( )

Метод *confirm(“текст”)* предназначен для вывода на экран пользователя диалогового окна с указанным сообщением и кнопками "ОК" и "Отмена".

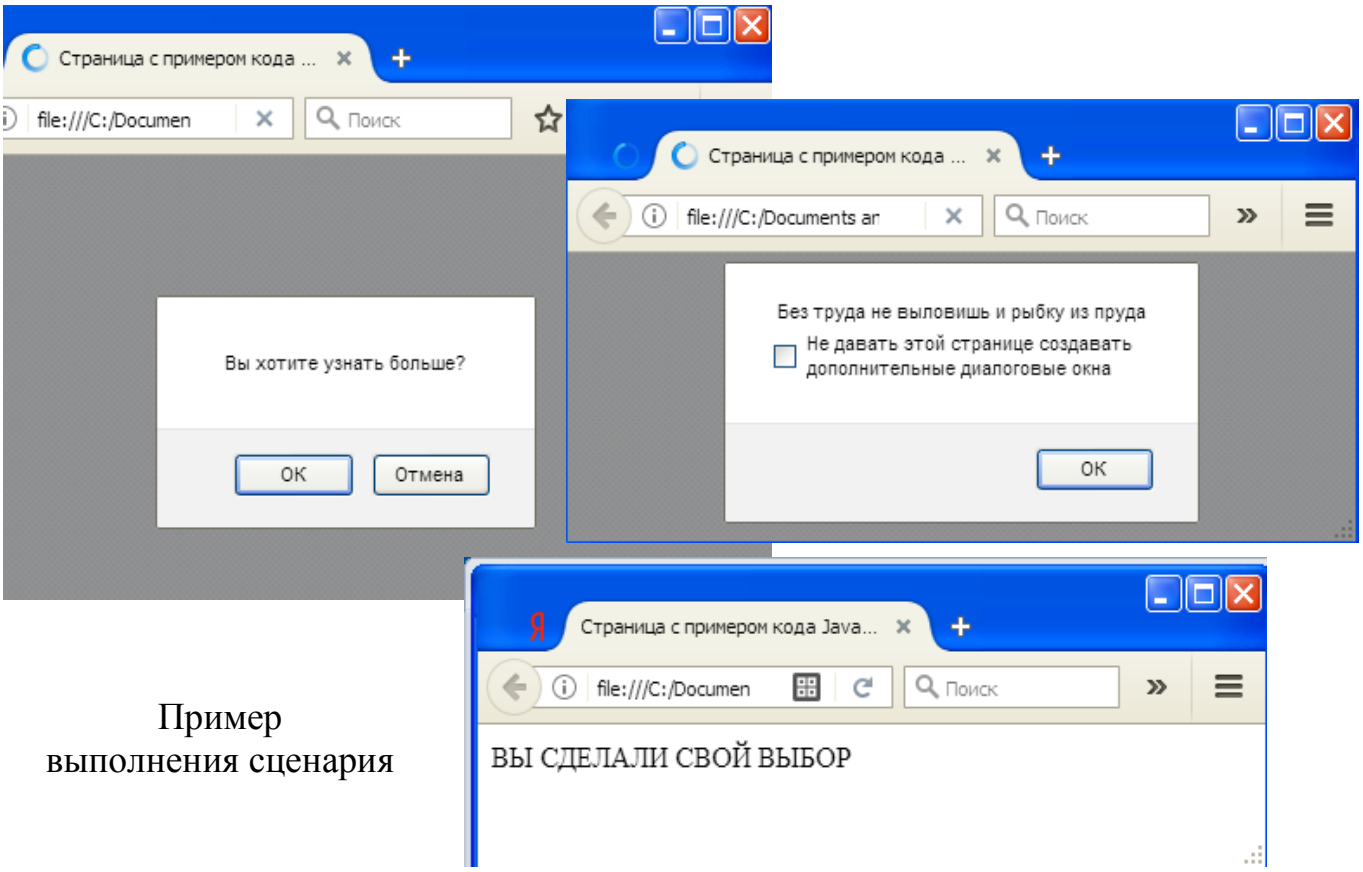
Результатом выполнения метода confirm() является одно из двух значений:

- ✓ *true*, если посетитель нажал "ОК";
- ✓ *false*, если посетитель нажал "Отмена" или закрыл окно диалога.

Окно с подтверждением используют, чтобы запросить у пользователя-посетителя разрешение на выполнение того или иного действия или, в зависимости от выбора, вывести одно из сообщений программиста.

Например, используя выше описанные методы, установите диалог с посетителем публикации, создав страницу со встроенным сценарием по образцу:

```
<html>
  <head>
    <title>Устанавливаем диалог</title>
    <SCRIPT>
        var User = confirm("Вы хотите узнать больше?");
        if (User) {
        alert("ВАМ придется приложить усилие. Вы готовы продолжить?");
        else {
        alert("Без труда не выловишь и рыбку из пруда");
        }
    </SCRIPT>
</head>
    <body>
    ВЫ СДЕЛАЛИ СВОЙ ВЫБОР
    </body>
</html>
```



Пример выполнения сценария

Метод prompt( )

Метод *prompt(“текст1”,”текст2”)* предназначен для вывода на экран пользователя диалогового окна с сообщением, <u>текстовым полем для ввода данных</u> и кнопками "ОК" и "Отмена".

“текст1” - сообщение, которое будет выведено в диалоговом окне. Параметр является обязательным;

”текст2” - является необязательным и может использоваться для указания начального значения, которое будет выведено в текстовое поле окна при открытии. Если при загрузки страницы текстовое поле должно быть пустым, то прописываются кавычки без содержания prompt(“текст1”,””).

В зависимости от действий посетителя метод prompt() возвращает данные:

текстовое значение - если в текстовое поле введены данные и посетитель нажал "ОК";

пустая строка - если посетитель не впечатал данные в текстовое поле и нажал "ОК";

null - если пользователь нажал "Отмена" или закрыл окно, при этом не важно, какие данные были введены в текстовое поле.

4.2 Внешний сценарий

Внешний сценарий JavaScript-кода оформляется отдельным файлом с расширением ***.js**

Потом при создании HTML-страницы в заголовке прописывается вызов скрипта с указанием полного имени файла-сценария:

```
<script src="имя_файла.js"></script>
```

Используя выше описанный метод вывода диалогового окна создайте в БЛОКНОТЕ файл под именем **2.js** с единственной строкой кода

```
alert("Приветствую ВАС, ДРУЗЬЯ!");
```

Создайте новый файл под именем с ниже приведенным HTML кодом:

```
<html>
    <head>
        <title>Первый опыт JavaScript</title>
            <script src="2.js"></script>
    </head>
    <body>
            Отличный результат!
    </body>
    </html>
```

Проверьте результат работы внешнего сценария, отрыв файл **3.html** в браузере.

Использование внешнего сценария с любым объемом программного кода позволяет выполнить скрипт в любой HTML - странице по вызову одним тегом.

Вопросы для самоконтроля

1) Что такое JavaScript?

2) Для чего используется сценарий JavaScript?

3) Какие типы сценариев вы знаете? Что такое встроенный и внешний скрипт?

4) Как оформляется в HTML – коде встроенный скрипт?

5) Какие методы используются для диалога с посетителем публикации?

6) В чем принципиальное отличие методов методы alert(), confirm(), prompt() друг от друга? Опишите синтаксис использования методов.

7) Как оформляется внешний скрипт?

8) Как использовать внешний скрипт в HTML- странице?

5 Теория построения программного кода JavaScript

5.1 Синтаксические особенности кода JavaScript

1) Основной единицей кода JavaScript является предложение.

2) **Предложение JavaScript** - это команда браузеру на выполнения конкретного действия.

3) Предложение заканчивается точкой с запятой "**;**".

Использование точки с запятой в конце предложения позволяет написание нескольких предложений в одной строке.

Отсутствие точки с запятой в конце строки в определенных случаях влияет на поведение кода, поэтому рекомендуется прописывать "**;**" для однозначного прочтения команды браузером.

4) **Код JavaScript** - это последовательность предложений JavaScript. Предложения выполняются браузером в том порядке, в котором они написаны.

5) **Блоки JavaScript** - это способ сгруппировать предложения вместе. **Блок** начинается открывающейся фигурной скобкой "**{**" и заканчивается закрывающейся "**}**".

Обычно блоки используются для выполнения нескольких команд в функции или внутри условной конструкции.

6) **Комментарии JavaScript** - строчки, отмеченные особым образом, которые не выполняет

Комментарии позволяют добавлять поясняющий текст в код программы или исключать некоторые куски кода для исполнения не удаляя их в процессе отладки сценария.

Комментарии бывают **однострочные** и **многострочные**.

7) **Однострочные комментарии** начинаются с двух слэшей "//" браузер не выполняет код от "//" до конца строки.

Пример использования однострочного комментария:

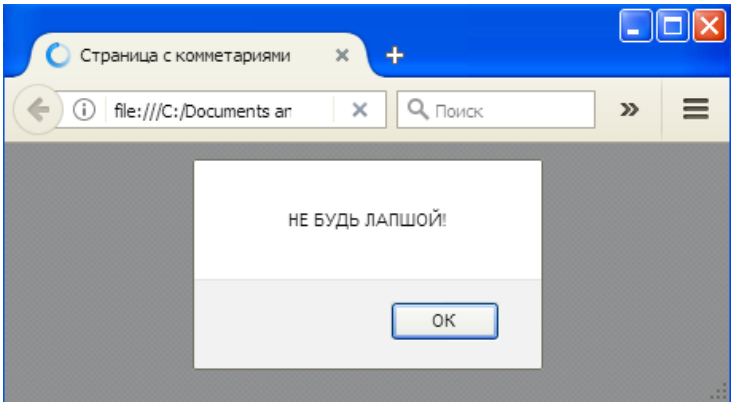
```
<script>
// Диалог с посетителем
alert("Здравствуйте!");   // Это приветствие в окне диалога
// alert("НЕ хочу УЧИТЬСЯ"); Эта команда не выполняется
</script>
```

8) **Многострочные комментарии** начинаются с одного слэша и звездочки "/*" и заканчиваются обратной комбинацией - звездочкой и слэшем "*/".

Все что находится между этими знаками, браузер будет считать комментарием и пропустит.

Например:

```
<script>
    /* Начало блока комментарий
    alert("Будь ЗДОРОВ!"); код не выполнится.
    alert("Расти БОЛЬШОЙ!"); код не выполнится.
    Конец блока комментарий */
    alert("НЕ БУДЬ ЛАПШОЙ!");
</script>
```



5.2 Переменные в JavaScript

JavaScript – объектно-ориентированный облегченный язык программирования**.** Как любой язык программирования JavaScript воспринимает и обрабатывает различные типы данных – строковые, числовые, логические.

Объект хранения данных называется **переменной**.

Каждую переменную:

1) **объявляют** – задают область использования,
2) **именуют** - присваивают имена, по которым к ним обращаются
3) **инициализируют -** присваивают переменной значение.

Переменные в JavaScript могут быть **Локальными** и **Глобальными**.

Локальные переменные - это переменные, **объявленные внутри функции** JavaScript. Доступны только в пределах той функции, внутри которой они

объявлены. При выходе из этой функции переменные уничтожаются, что позволяет объявлять внутри разных функций переменные с одинаковым именем - они никак не будут пересекаться, поскольку используются только внутри функции, в которой они созданы.

Глобальные переменные объявляются вне функций и к ним могут обращаться все функции и скрипты на странице. Уничтожаются глобальные переменные при закрытии страницы.

1) Объявление переменных

Переменную в JavaScript перед использованием необходимо создать - объявить.

Объявление переменной выполняется с помощью ключевых слов

var или *let*.

Например:

```
var x;
var Name;
let S;
let Kot;
```

Каждая строка объявления переменной является отдельным предложением кода JavaScript и оканчивается точка с запятой.

Если переменную объявить без использования ключевого слова "var", то она автоматически объявляется глобальной, даже если объявление произведено внутри функции.

Например выражения s = 25 объявят переменную s как глобальную.

Объявление с помощью ключевого слова **let** имеет несколько важных отличий от объявления с помощью ключевого слова **var:**

- область видимости. При объявлении через **var** переменная работает внутри всей функции, в которой произошло объявление. Переменная объявленная словом **let** видна только внутри блока **{ ... }**, в котором она объявлена;

- видимость по времени. Переменная, объявленная ключевым словом **let** видна только после объявления, а переменная, объявленная ключевым словом **var** может быть доступна в коде и до того места, в котором она объявлена;

- при использовании переменной в качестве счетчика цикла, объявленная через слово **var** переменная живет в течение всего выполнения цикла и доступна даже после его завершения. При объявлении через **let**, каждой итерации цикла будет соответствовать своя независимая переменная.

2) Инициализация переменной (присваивание значения)

Присвоить значение переменной, или **инициализировать** ее можно на этапе объявления:

```
var c = 1;
let Name = "Ivan";
```

Присвоение значения уже объявленной переменной:

```
x = 11;
student = "КСК";
```

Знак "=" - операция присвоения, а никак не логическая операция сравнения – «равенство» "==".

В процессе выполнения операции присваивания значение выражения, расположенного справа от знака "=" присваивается переменной, расположенной слева от знака "=".

3) Именование переменной

Правила именования переменных JavaScript:

1) при именовании переменной важно учитывать регистр букв (X и x – разные переменные);

2) имена переменных должны начинаться с буквы или символов "$" и "_";

3) имя переменной составляется из любых цифр и букв **латинского алфавита**, символов "$" и "_";

4) в качестве имени переменной нельзя использовать зарезервированные и ключевые слова.

Задание для самостоятельного выполнения

Используя методы alert(), confirm(), prompt(), операторы ветвления и выбора создайте сценарий – «тестирование», установив диалог с посетителем:

1) выводится вопрос: «Назовите 9 месяц года». Посетитель должен ввести название месяца и увидеть оценку своих знаний;

2) выводится определение известного понятия (термина). Посетитель должен ввести название и увидеть оценку своих знаний;

3) выводится окно с вопросом по математике (например, «Рассчитайте и впечатайте результат вычисления 1045-659»). В зависимости от введенного ответа посетитель должен увидеть комментарий, оценивающий его способности к математике.

Вопросы для самоконтроля

1) Что такое переменная в JavaScript?

2) Для чего используется переменная при написании сценария JavaScript?

3) Поясните понятия глобальной и локальной переменной.

4) Перечислите основные правила работы с переменными.

5) Для чего используются слова *var* и *let* ?

6) В чем состоит принципиальная разница объявления переменных словом *var* от объявления переменных словом *let* ?

7) Поясните правила именования переменной в JavaScript?

8) Объясните суть применения знака “=”. В какой операции используется “=”, а в какой “==”?

9) Поясните правила и приведите примеры присваивания значения переменной.

Приложение А. Примеры кода сценария JavaScript

Пример №1.

Сценарий вывода результата вычисления суммы двух чисел на Web-страницу:

```
<!DOCTYPE html>
    <head>
        <title>Первый пример JavaScript</title>
    </head>
<body>
        <h1>Результат вычисления суммы 123+96 </h1>
        <center>
            <font size=7 color="red">
              <b>
               <script>
                    document.write(123 + 96);
               </script>
              </b>
            </font>
        </center>
</body>
</html>
```

Пример №2.

Использование модального окна для вывода результата математической операции.

```
<!DOCTYPE html>
    <head>
        <title>Второй пример JavaScript</title>
    </head>
<body>
        <h1>Результат вычисления суммы 123+96 </h1>
               <script>
                    window.alert(123 + 96);
               </script>
</body>
</html>
```

Пример №3.

Встроенный сценарий JavaScript в код HTML - страницы:

(создается функция пользователя – одноразовая операция изменения текста по клику на кнопку)

```
<!DOCTYPE html>
<html>
<head>
    <title>Третий пример JavaScript</title>
     <script>
     function myFunction() {
document.getElementById("demo").innerHTML = "Ученье - СВЕТ";
     }
     </script>
</head>
<body>
<center>
  <h1>Использование встроенного скрипта </h1>
      <b>
         <font size=7 color=”red”>
          <p id="demo">Хотите знать НАЧАЛО?</p>
            <button type="button" onclick="myFunction()">
                 А НЕУЧЕЙ – ТЬМА
            </button>
         </font>
      </b>
</center>
</body>
</html>
```

Пример №4

Манипулирование CSS: *изменение фона страницы*

```
<!DOCTYPE html>
<html>
<head>
    <title>Четвертый пример JavaScript</title>
<script>
        function bgChange(bg) {
                document.body.style.background = bg;
                }
</script>
</head>

<body>
<center>
<h1>Поиграем в светофор</h1>
<h2>Наведите курсор на цветную ячейку!</h2>

<table style="width:300px;height:100px">
   <tr>
      <td onmouseover="bgChange(this.style.backgroundColor)"
           onmouseout="bgChange('transparent')"
           style="background-color:RED">
      </td>
      <td onmouseover="bgChange(this.style.backgroundColor)"
           onmouseout="bgChange('transparent')"
           style="background-color:Gold">
      </td>
      <td onmouseover="bgChange(this.style.backgroundColor)"
           onmouseout="bgChange('transparent')"
           style="background-color:Green">
      </td>
   </tr>
</table>
</center>
</body>
</html>
```

Перечень рекомендуемых источников

для самостоятельной работы студентов

1) http://forcoder.ru/javascript/
2) https://learn.javascript.ru/
3) http://www.w3im.com/ru/js/js_examples.html
4) http://labs.org.ru/567-2/javascript/