ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ЦИФРОВЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

Кафедра «Информационные технологии»

Практикум по дисциплине «Клиентские интернет-технологии»

Составители
Борисова Е.В.
Зубарева Е.Г.
Цынко Д.О.

Ростов-на-Дону, 2023

Аннотация

Практикум предназначен для студентов всех форм обучения направления 09.03.03 «Прикладная информатика» и 09.03.02 «Информационные системы и технологии».

Практикум соответствует требованиям, предъявляемым Федеральным государственным образовательным стандартом высшего образования.

Составители

доцент к.т.н. кафедры
«Информационные технологии» Борисова Е.В.

старший преподаватель кафедры
«Информационные технологии» Зубарева Е.Г.

программист кафедры
«Информационные технологии» Цынко Д.О.

Оглавление

ВВЕДЕНИЕ

Практикум предназначен для студентов, изучающих дисциплину «Клиентские интернет-технологии».

В результате изучения дисциплины, студенты познакомятся с:

- технологиями и основными принципами объектно-ориентированного программирования;
- принципами создания динамических Web-документов;
- основными элементами языка;
- взаимосвязью языков скриптов и таблицей стилей для оформления Web-документов;
- организацией проверки данных введенных пользователем.

По окончании изучения данной дисциплины студенты смогут:

- создавать Web-документы с динамически изменяемым содержимым;
- использовать стилевое форматирование совместно с языками сценариев для расширения возможностей оформления документов.
- использовать технологии объектно-ориентированного программирования, необходимые для Web-разработки;

В результате освоения данного практикума обучающиеся должны знать:

– принципы построения web-приложений;

– языки разметки и программирования для web;

понимать:

– чужой JavaScript-код;

– как разрабатывать одностраничные приложения;

Уметь продемонстрировать:

– работу с инструментами разработки для web;

– работу с инструментами отладки и профилирования JavaScript.

ОБЩИЙ ОБЗОР ЯЗЫКА

Основные определения

Любая программа оперирует некими данными: именем стилевого класса, размерами элемента, цветом шрифта и прочие. JavaScript может манипулировать данными, относящимися к разным типам.

Тип данных описывает их возможные значения и набор применимых к ним операций. Типы данных бывают простыми и сложными. Сущность, относящаяся к простому типу данных может хранить только одно значение (это строковые, числовые и логические типы данных). Сущность сложного типа данных может хранить сразу несколько значений. Например – массивы. Другой пример сложного типа данных – объекты.

Для создания механизма управления страницами на клиентской стороне было предложено использовать объектную модель документа. Суть модели в том, что каждый HTML-контейнер - это объект, который характеризуется тройкой:

- Свойства
- Методы
- События

Существование программных объектов самих по себе не имеет никакого смысла. Они дадут преимущества при программировании тогда, когда можно организовать их взаимодействие.

Язык ядра JavaScript

Синтаксис языка

Язык JavaScript чувствителен к регистру. Приложение JavaScript представляет собой набор операторов языка (команд), последовательно обрабатываемых встроенным в браузер интерпретатором. Каждый оператор можно располагать в отдельной строке. В этом случае разделитель ‘;’, отделяющий один оператор от другого, не обязателен. Его используют только в случае задания нескольких операторов на одной строке. Любой оператор можно расположить в нескольких строках без всякого символа продолжения. Например, следующие два вызова функции alert эквивалентны:

```
… alert("Подсказка");
alert(
"Подсказка"
);
…
```

Нельзя перемещать на другую строку единый строковый литерал - он должен располагаться полностью на одной строке текста программы или разбит на два строковых литерала, соединенных операцией конкатенации ‘+’:

```
…
alert("Подсказка"); // правильно
alert("Под
сказка");          // не правильно
alert("Под" +
"сказка");      // правильно (но браузер выведет текст одной строкой!)
…
```

Пробельные символы в тексте программы являются незначащими, если только они не используются в строковых литералах.

В JavaScript строковые литералы можно задавать двумя равноправными способами - последовательность символов, заключенная в двойные или одинарные кавычки:

```
"Анна"
'Анна'
```

В строковых литералах можно использовать ESC-последовательности, которые начинаются с символа обратной наклонной черты, за которой следует обычный символ.

Некоторые подобные комбинации трактуются как один специальный символ (таб.1).

Таблица 1.

Esc-последовательности	Символ
\b	Возврат на один символ
\f	Переход на новую страницу
\n	Переход на новую строку
\r	Возврат каретки
\t	Горизонтальная табуляция Ctrl-I
\'	Апостроф
\"	Двойные кавычки
\\	Обратная наклонная черта

ESC-последовательности форматирования используются при отображении информации в диалоговых окнах, отображаемых функциями alert(), prompt() и confirm(), а также, если методом document.write() записывается содержимое элемента pre.

Комментарии в программе JavaScript двух видов - однострочные и многострочные:

// комментарий, расположенный на одной строке.

/*

комментарий, расположенный на нескольких строках.

***/**

Ссылка на объект осуществляется по имени, заданному параметром name тэга HTML, с использованием точечной нотации. Например, пусть в документе задана форма с двумя полями ввода:

<form name="form1">

Фамилия: <input type = "text" name = "student" size = 20>

Курс: <input type = "text" name = "course" size = 2>

</form>

Для получения фамилии студента, введенного в первом поле ввода, в программе JavaScript следует использовать ссылку document.form.student.value, а чтобы определить курс, на котором обучается студент, необходимо использовать ссылку document.form.course.value.

Переменные и литералы в JavaScript

В JavaScript все переменные вводятся с помощью одного ключевого слова var. Синтаксическая конструкция для ввода в программе новой переменной с именем name1 выглядит следующим образом:

var name1;

Объявленная таким образом переменная name1 имеет значение 'undefined' до тех пор, пока ей не будет присвоено какое-либо другое значение, которое можно присвоить и при ее

объявлении:

```
var name1 = 5;
var name1 = "новая строковая переменная";
```

JavaScript поддерживает четыре простых типа данных:

- Целый
- Вещественный
- Строковый
- Логический (булевый)

Для присваивания переменным значений основных типов применяются литералы – буквальные значения данных соответствующих типов.

Выражения JavaScript

Выражение – комбинация переменных, литералов и операторов, в результате вычисления которой получается одно единственное значение. Переменные в выражениях должны быть инициализированы.

1. Присваивание

Оператор присваивания (=) рассматривается как выражение присваивания, которое вычисляется равным выражению правой части, и в то же время он присваивает вычисленное значение выражения переменной, заданной в левой части:

```
var name2=10;
```

2. Арифметическое выражение

Вычисляемым значением арифметического выражения является число. Создаются с помощью арифметических операторов (таб.2).

Таблица 2.

Оператор	Действие
+	Сложение
-	Вычитание
*	Умножение
/	Деление
%	Остаток от деления целых чисел
++	Увеличение значения на единицу
--	Уменьшение значения на единицу

3. Логическое выражение

Вычисляемым значением логического выражения может быть true или false. Для создания используются операторы сравнения или логические операторы, применяемые к переменным любого типа (таб.3).

Таблица 3.

Операторы сравнения	**Значение**	**Логические Операторы**	**Значение**
= =	Равно	&&	логическое И
! =	Не равно	||	логическое ИЛИ
>=	Больше или равно	!	логическое НЕ
<=	Меньше или равно		
>	Строго больше		
<	Строго меньше		

4. Строковые выражения

Вычисляемым значением строкового выражения является число. В JavaScript существует только один строковый оператор – оператор конкатенации (сложения) строк:

string1 = “Моя ” + “строка”

Стандартные объекты и функции ядра JavaScript

Объект Array

Массив - упорядоченный набор однородных данных, к элементам которого можно обращаться по имени и индексу. Язык JavaScript не имеет встроенного типа данных для создания массивов, поэтому для решения используется объект Array и его методы.

Для создания объекта Array вызывается оператор new и конструктор массива - системная функция (ее имя совпадает с именем объекта), инициализирующая элементы массива:

```
m=new Array() ;
```

Заполнение массива происходит позже. Например :

```
<script language="JavaScript">
//создание нового массива
m=new Array();
//заполнение массива
m[0]=1;
m[1]=2;
m[2]=4;
m[3]=56;
</script>
```

В приведенном выше примере с помощью команды new создается массив m, а затем происходит его заполнение - каждому элементу присваивается определенное значение.

```
m=new Array(1,2,4,56);
```

Вызывается команда new и сразу задаются значения всех элементов массива.

```
<script language="JavaScript">
//создание нового массива и его заполнение
m=new Array(1,2,4,56)
</script>
```

Объявление строковых массивов проводится тем же способом, что и объявление числовых массивов.

Таблица 4.

Методы объекта Array	Действие
join()	Объединяет все элементы массива в одну строку с указанием разделителя.

reverse()	Изменяет порядок элементов в массиве - первый элемент становится последним, последний - первым
sort()	Выполняет сортировку элементов массива
split()	Разделяет строку на составные части
concat()	Объединяет два массива в один
slice()	Выделяет часть массивы
toString()	Возвращает строку элементов массива. разделены запятой.
Свойство length	Возвращает длину массива (число элементов в нем).

Пусть определены два массива:

array1 = new Array("Первый","Второй","Третий");

array2 = new Array("Один","Два","Три");

Тогда метод join() первого массива array1.join() возвратит строку: "Первый,Второй,Третий".

Метод sort() первого массива array1.sort() упорядочит элементы массива array1 (переставив их местами непосредственно в самом массиве array1) в алфавитном порядке:

array1[0] = "Второй"; array1[1] = "Первый"; array1[2] = "Третий";

Поскольку некоторые методы массива возвращают массив, то к нему можно сразу же применить какой-либо метод, продолжив "точечную" нотацию. Например, array1.concat(array2).sort() объединит два массива в один новый и отсортирует его.

Объект Date

Используется для представления дат в программах JavaScript. Время храниться в виде числа миллисекунд, прошедших от 1 января 1970 года.

Объект document

Объект document имеет дело прежде всего с телом HTML-страницы. Он имеет несколько дочерних объектов (коллекций): all, images, link, anchor и form. Пользуясь объектной моделью построения документа можно, например, обратиться к любой картинке на странице через следующий синтаксис:

document.images.name.src

Для document не существует никаких событий. Некоторые свойства и методы перечислены в таблице, из методов наиболее употребимы write и writeln (таб.5).

Таблица 5.

Методы	Назначение
write() **writeln()**	Записывает HTML-текст в текущий документ и должен вызываться, когда документ открывается для записи. Синтаксис: document.write('somestring'), где somestring может быть одной строкой, переменной или же несколькими связанными строками в формате HTML, которые выводятся на экран
lastModified()	Показывает дату последней модификации документа: date1 = document. lastModified
open()	Открывает документ для записи дополнительных строк в формате HTML: document.open()
close()	Закрывает документ, который вызывался методом document.open(): document.close().

Таблица 6.

Свойства	Назначение
bgColor	Устанавливает цвет фона текущего документа. Этот цвет может иметь шестнадцатеричное представление #rrggbb или соответствующее название. Синтаксис: document.bgColor="#e7e6d8"
fgColor	Устанавливает цвет текста документа. Аналогичен по функциям свойству bgColor
referrer	Указывает url документа, на который ссылается пользователь в настоящее время. Например, если кто-то обратился по адресу: http://www.nm.org/welcome.htm с сервера

Данный объект создается также, как и любой объект в JavaScript – с помощью оператора new и конструктора, в данном случае Date():

date1 = new Date(); // значением переменной date1 будет текущая дата Параметром конструктора может быть строка, в которой записана нужная дата:

date1 = new Date("january 14, 2000, 12:00:00");

Можно задать список параметров: date1 = new Date(2000, 1, 14, 12, 0, 0);

Объект Math

В свойствах данного объекта хранятся основные математические константы, а его ме-

тоды вычисляют основные математические функции. При обращении к данному объекту, создавать его не надо, но необходимо явно указывать его имя Math. Например:

```
p = Math.PI;        // хранится значение числа пи.
```

Объект String

Можно явно создавать строковый объект, используя оператор new и конструктор:

```
myString = new String("Hello!");
```

Данный объект имеет единственное свойство length, хранящее длину строки, содержащейся в строковом объекте, и два типа методов: одни непосредственно влияют на содержание самой строки, вторые возвращают отформатированный HTML-вариант строки.

Стандартные функции верхнего уровня

В JavaScript существуют несколько функций, для вызова которых не надо создавать никакого объекта, она находятся вне иерархии объектов.

Функция parseFloat(parameter) анализирует значение переданного ей строкового параметра на соответствие представлению вещественного числа.

Функция parseInt(parameter, base) пытается возвратить целое число по основанию, заданному вторым параметром.

Эти функции полезны при анализе введенных пользователем данных в полях формы до их передачи на сервер.

Функции Number(object) и String(object) преобразуют объект, заданный в качестве его параметра в число или строку.

Объекты клиента

При интерпретации страницы HTML браузером создаются объекты JavaScript, свойства которых представляют значения параметров тэгов языка HTML.

Иерархия объектов

Созданные объекты существуют в виде иерархической структуры, отражающей структуру самой HTML-страницы. На верхнем уровне расположен объект window, представляющий собой активное окно браузера. Далее вниз по иерархической лестнице следуют объекты frame, document, location и history и т.д.

Значения свойств объектов отражают значения соответствующих параметров тэгов стра-

ницы или установленных системных параметров. На рисунке показана структура объектов клиента (браузера).

Особняком стоит объект navigator с двумя дочерними (подчиненными) объектами. Он относится к самому браузеру, и его свойства позволяют определить характеристики программы просмотра. Каждая страница в добавление к объекту navigator обязательно имеет еще четыре объекта:

- window — объект верхнего уровня, свойства которого применяются ко всему окну, в котором отображается документ;
- document — свойства которого определяются содержимым самого документа: связи, цвет фона, формы и т. д.;
- location — свойства которого связаны с url-адресом отображаемого документа;
- history — представляет адреса ранее загружавшихся HTML- страниц.

Объект navigator

Этот объект применяется для получения информации о версиях. Синтаксис: navigator.name_properties

Методы и события, догадаться не определены для этого объекта. Да и свойства только для чтения, так как ресурс с информацией о версии недоступен для редактирования.

Свойства:

- appCodeName - кодовое имя браузера;
- appName - название браузера;
- appVersion - информация о версии браузера;
- userAgent - кодовое имя и версия браузера.

Ниже приведен пример использования объекта navigator.

```
<HTML>
<HEAD>
<title>Test of Browser name</title>
</HEAD>
<BODY>
<h1 align=center>Проверка имени типа браузера;</h1>
<hr>
Для того, чтобы получить имя вашей программы просмотра
выберите кнопку "Browser"<br>
<center>
```

```
<form name=kuku>
<input type=button name=browser value=Browser
 onClick="window.alert(window.navigator.appName)">
</form>
</BODY>
</HTML>
```

Объект window

Объект window создается автоматически при запуске браузера, так как для отображения документа необходимо окно. Одно из назначений объекта окна - это создание нового окна. Новое окно браузера создается с помощью метода window.open(). Метод window.open() имеет ряд дополнительных аргументов, которые позволяют задать местоположение окна, его размер и тип, а также указывают, должно ли окно иметь полосы прокрутки, полосу команд и т. п. Помимо этого можно задавать и имя окна.

В общем виде данный метод можно представить следующим образом: window.open('url', 'name', 'parameters')

Рассмотрим синтаксис более подробно:

- Первый параметр метода window.open() - это url документа, загружаемого в окне. Если его не заполнить, то окно останется пустым.
- Второй параметр определяет название окна (name). Это имя может использоваться для обращения к созданному окну.
- Третий параметр представляет список необязательных опций, разделенных запятой. С их помощью Вы определяете вид нового окна: наличие в нем панелей инструментов, строки состояния и других элементов.

Приведем таблицу с описанием параметров нового окна, задаваемого третьим параметром (parameters) метода open().

Объект window использует три метода отображения сообщений:

- метод prompt() – выводит диалоговое окно с полем ввода, куда пользователь может ввести информацию
- метод alert() – выводит на экран окно - сообщение с кнопкой ОК и определенным программистом текстом
- метод confirm() – выводит диалоговое окно с кнопками ОК и Cancel. Дает возможность пользователю продолжить или отменить предложенную операцию.

Сообщение, которое вы хотите вывести на экран, набирается в кавычках внутри круглых скобок.

Данный скрипт запрашивает имя посетителя и выдает приветствие с введенным именем.

```
<script language="JavaScript">
 name=window.prompt ("Введите, пожалуйста, свое имя", "Ваше имя"); window.alert
 ("Вас зовут, " + name);
</script>
```

В этом фрагменте кода метод prompt имеет следующие параметры: текст запроса и значение, заполняющее поле ввода по умолчанию; переменная name - имя переменной, куда сохраняется введенная информация (имя может быть любым).

Объединение JavaScript и CSS

1. Пример изменения цвета текста.

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<h1 style="color:red">Добро пожаловать на нашу страницу!</h1>
<p>Здесь много интересной информации. Здесь много
      интересной информации. Здесь много интересной
      информации. Здесь много интересной информа-
      ции. Здесь много интересной информации. </p>
      </body>
</html>
```

Данный пример применения CSS позволяет сделать заголовок красного цвета. Допустим, вы хотите, чтобы текст заголовка только тогда становился красным, когда пользователь наводит на него курсор. Этого можно добиться с помощью CSS и JavaScript.

Шаг 1. Удаление существующей информации о стиле

Это действие может показаться вам шагом назад, но оно действительно необходимо:

```
<html>
<head>
      <title>Простая страница</title>
</head>
<body>
<h1>Добро пожаловать на нашу страницу! </h1>
<p>Здесь много интересной информации. Здесь много
      интересной информации. Здесь много интересной
      информации. Здесь много интересной информа-
      ции. Здесь много интересной информации. </p>
</body>
</html>
```

Шаг 2. Добавление идентификатора

Поскольку вам нужно как-то обращаться к элементу, с которым будут производиться манипуляции, необходимо в тэг <hl> добавить атрибут id - это краткое обозначение, позволяющее указать нужный элемент.

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<hl id="head1">Добро пожаловать на нашу страницу!</h1>
<p>Здесь много интересной информации. Здесь много
    интересной информации. Здесь много интересной
    информации. Здесь много интересной информа-
    ции. Здесь много интересной информации. </p>
    </body>
</html>
```

Шаг 3. Добавление обработчика событий

Следующий шаг — добавление обработчика событий. Этому действию соответствует событие onMouseover. Также следует указать имя функции, которая будет вызываться при выполнении события:

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<h1 id="head1" onMouseover ="colorchange ()">Добро пожаловать на нашу
страницу!</h1>
<p>Здесь много интересной информации. Здесь много
    интересной информации. Здесь много интересной
    информации. Здесь много интересной информа-
    ции. Здесь много интересной информации. </p>
</body>
</html>
```

Шаг 4. Написание сценария JavaScript

Вам потребуется единственная строка, состоящая из следующих частей:

- имя объекта на странице, с которым должен выполняться ваш сценарий - в данном случае head1;
- применяемый аспект JavaScript - в данном случае style;
- атрибут стиля, который будет изменяться - color;
- новое значение, принимаемое атрибутом стиля - red. Соедините это, и получится следующая строка: Head1.style.color = "red"

Добавьте ее в функцию и сохраните файл. В окончательном варианте страница должна

выглядеть так:

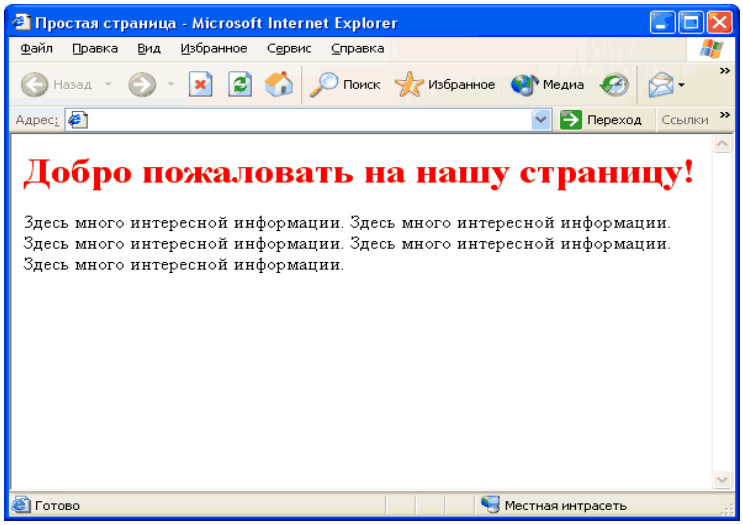


Рисунок 1 – Пример работы кода

Используя в сценариях JavaScript атрибуты, название которых пишется через дефис, убирайте дефис и пишите оба слова слитно, причем второе слово должно начинаться с заглавной буквы. Таким образом, text- decoration в сценариях должно выглядеть как textDecoration.

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript"> func-
tion addunderline(){
head.style.textDecoration = "underline";}
function removeunderline(){
head.style.textDecoration = "none";}
</script>
</head>
<body>
<h1 id="head" onMouseover="addunderline()" onMouseout="removeunder-
line()">Добро пожаловать на нашу страницу! </h1>
<p>3десь много интересной информации. Здесь много
      интересной информации. Здесь много интересной
      информации. Здесь много интересной информа-
      ции. Здесь много интересной информации.</p>
</body>
</html>
```

Необходимо обратить внимание на прописную букву в слове textDecoration — если все слово набрать в нижнем регистре, сценарий выполняться не будет. Теперь при наведении курсора заголовок станет подчеркнутым, а затем, если убрать курсор, вернется в прежнее состояние.

2. Пример точного позиционирования текста

Сначала необходимо рассмотреть, каким образом осуществляется позиционирование

текста.

Наиболее часто применяются следующие атрибуты позиционирования:

- position - имеет два интересующих нас значения: absolute и relative (по умолчанию значение static). Для значения absolute в качестве точки отсчета используется верхний левый угол окна браузера, и все параметры местоположения отмеряются от него. В свою очередь, для relative точкой отсчета является то место, в котором разместился бы элемент страницы, если бы не было представлено никакой информации о местоположении;
- top - используется для указания вертикального смещения элемента от точки отсчета. Величина смещения может выражаться в различных единицах (пикселы, дюймы, сантиметры, миллиметры и т.п.). В наших примерах используются пикселы. Положительное значение top соответствует смещению элемента страницы вниз, в то время как отрицательное - по направлению к верхней границе окна браузера;
- left - подобен атрибуту top, но применяется для указания горизонтального направления. Положительное значение соответствует сдвигу элемента вправо, отрицательное - влево.

```
<html>
<head>
<title>Простая страница</title>
</head>
<body>
<h1 style=" text-decoration: underline"> Добро пожаловать на нашу страницу!</h1>
<p style="position:absolute; top:125px; left:200px">Простой текст для позиционирования.</p>
</body>
</html>
```

С помощью CSS текст расположен со значением absolute, то есть его положение отсчитывается от верхнего левого угла окна браузера. Значение атрибута top равно 125px, таким образом, текст будет расположен на 125 пикселов ниже верхнего края страницы. Значение атрибута left равно 200px, то есть текст будет сдвинут на 200 пикселов от левого края окна браузера.

Лабораторная работа №1. Размещение скриптов в HTML-документе

1.1 Задание 1

1. Создайте простой HTML-документ.

2. Добавьте два абзаца с произвольным текстом.

3. Организуйте между двумя абзацами вывод приветственного сообщения в диалоговом окне, задав необходимые команды внутри тэга <script>.

4. Добавьте команду вывода аналогичного приветственного сообщения в окно браузера после закрытия диалогового окна.

5. Сохраните документ с именем Ex1.html в рабочей папке.

1.2 Задание 2

1. Создайте простой HTML-документ.

2. Добавьте два абзаца с произвольным текстом.

3. Организуйте между двумя абзацами вывод приветственного сообщения в диалоговом окне, задав необходимые команды JavaScript во внешнем файле. Для этого:

• создайте новый текстовый файл,

• поместите в него код JavaScript,

• сохраните файл с именем main.js следующим образом: укажите тип файла “Все файлы”, кодировку “UTF-8”.

4. Добавьте ссылку на внешний скриптовый файл из рабочего HTML- документа.

5. Сохраните документ с именем Ex2.html в рабочей папке.

1.3 Задание 3

1. Создайте простой HTML-документ.

2. Сохраните документ с именем Ex3.html в рабочей папке.

3. Добавьте в документ код JavaScript так, чтобы в диалоговом окне появлялось поле с надписью "Введите сюда своё имя" и со значением по умолчанию в поле "Введите имя". Для этого используйте метод promt(...) объекта window. Для хранения введенного значения заведите новую переменную.

4. Организуйте вывод введенного значения имени в окно браузера в виде: "Ваше имя <…..>".

5. Дополните код, чтобы в новом диалоговом окне появилось надпись "Начать заново? " При положительном ответе появлялось диалоговое окно: "Не надоело? ", при отказе – "Ну и правильно!". Используйте для написания методы alert(…) и confirm(…) объекта window.

Лабораторная работа №2. Операторы управления, функции. Объекты ядра JavaScript.

2.1 Задание 4

1. Рассмотрите пример скрипта:

```
<html>
<head>
<title>if</title>
</head>
<body>
<script language="JavaScript" type="text/JavaScript">
var x, y;
x=parseInt(prompt("Введите значение x",")); // метод parseInt()
переводит строку в целое
y=parseInt(prompt("Введите значение y",")); // число
if(x<y){
alert("Максимальное число - y")}
else {
alert("Максимальное число - x")}
</script>
</body>
</html>
```

2. Допишите скрипт так, чтобы при введении пользователем одинаковых чисел, открывалось сообщение "Введенные числа равны!".

3. Напишите скрипт, в котором пользователя просят ввести правильный пароль. При вводе правильного пароля, в окне браузера появляется сообщение о том, что пароль верен. При вводе неправильного пароля – выпадает сообщение о неправильно введенном пароле. Для выполнения задания введите переменную password, в которую сохраните верное значение пароля.

4. Сохраните документ с именем Ex4.html в рабочей папке.

2.2 Задание 5

1. Рассмотрите пример скрипта:

```
<html>
```

```
<head>
<title>for</title>
</head>
<body>
<h1>Пример простой</h1>
<script language="JavaScript" type="text/JavaScript">
function line() {
document.writeln("<hr align='center' width='100'>");
}
for (var i=1; i<10; i++)
line();
</script>
</body>
</html>
```

2. Создайте вариант прорисованных линий со следующим условием:

• десять линий должны располагаться друг под другом,

• первая должна быть длинной 10 пикселей,

• каждая последующая на 10 пикселей больше.

3. Сохраните документ с именем Ex5.html в рабочей папке.

2.3 Задание 6

1. Создайте простой HTML-документ.

2. Сохраните документ с именем Ex6.html в рабочей папке.

3. Добавьте в документ код JavaScript так, чтобы в окне браузера была выведена таблица степеней двойки вида:

Степень Результат

2^0 1

2^1 2

2^2 4

2^3 8

2^4 16

2^5 32

Для этого в сценарии используйте метод write(…) объекта document для формирования содержимого страницы. На каждой итерации цикла for сформируйте очередную строку таблицы, в первую ячейку которой заносится соответствующая степень двойки, а во вторую результат ее возведения в указанную степень. Для выполнения этого действия используется

встроенный объект Math и его метод pow(…), возводящий первый параметр в степень, заданную вторым параметром. Обратите внимание, что метод write(…) может вызываться с любым количеством фактических параметров. Результатом его работы в любом случае является вывод в документ строки, полученной конкатенацией всех параметров, переданных в метод.

2.4 Задание 7

1. Рассмотрите пример скрипта:

```
<html>
<head>
<title>array</title>
</head>
<body>
<script language="JavaScript">
year=new Array("декабрь","январь","февраль","март","апрель","май",
"июнь","июль","август","сентябрь","октябрь","ноябрь");
summer=new Array(); //летние месяцы
summer=year.slice(6,9);
document.write(summer+"<br>");
</script>
</body>
</html>
```

2. Создайте массив, содержащий названия школьных предметов. Выделите из него два массива. Пусть к первому относятся предметы из раздела точных наук, а ко второму - из раздела гуманитарных наук. Для создания и вывода в окно браузера новых массивов используйте метод slice(…) и wtite(…) объекта document. Оформите исполняющий скрипт в виде отдельной функции, описанной в разделе <head> и вызванной в разделе <body>.

3. Сохраните документ с именем Ex7.html в рабочей папке.

2.5 Задание 8

1. Создайте простой HTML-документ.

2. Сохраните документ с именем Ex8.html в рабочей папке.

3. Добавьте скрипт, на основе которого будут выполняться следующие условия:

- если на страницу зашел пользователь через браузер Microsoft Internet Explorer, перенаправьте его автоматически на страницу Ex1.html;
- если на страницу зашел пользователь через любой другой браузер, перенаправьте его на страницу Ex3.html.

Для выполнения задания используйте свойство appName объекта navigator.

```
<html>
<head>
<title>Ex8</title>
</head>
<body>
  <script type='text/javascript'>
  a=navigator.appName;
  if (a=="Microsoft Internet Explorer"){
  location.href='Ex1.htm';
  } else {
  location.href='ex3.htm';
  }
  </script>
</body></html>
```

Лабораторная работа №3. Объекты клиентских приложений. Обработка событий

3.1 Задание 9

1. Рассмотрите скрипт:

```
<html>
<head>
<title>document</title>
</head>
<body>
<script language="JavaScript" type="text/JavaScript">
document.write("Спасибо, что пришли к нам на курсы!");
</script>
</body>
</html>
```

2. Допишите скрипт так, чтобы
   - цвет фона документа был #E7E6D8,
   - цвет шрифта – красный,
   - внизу выводилась дата последней модификации документа,
   - используйте для этого слияние методов wtite(…) и lastModified(…) объекта document.
3. Сохраните документ с именем Ex9.html в рабочей папке.

3.2 Задание 10

1. Рассмотрите пример скрипта открытия нового окна на странице:

```
<html>
<head>
<title>window</title>
</head>
<body>
<h1>Создание нового окна</h1>
<hr>
<script language="JavaScript" type="text/JavaScript">
window.open("http://www.google.com","","toolbar=no,scrollbars=yes,width=250, height=250, resizable=yes, top=100, left=500")
</script>
```

</body>

</html>

2. Измените скрипт так, чтобы выполнялись следующие условия:

- открытие нового окна происходило при нажатии на ссылку с текстом: «Щелкните на ссылке для получения справочной информации»,
- размеры окна - 500x500,
- есть возможность изменения размеров окна.

Для выполнения задания используйте написание функции.

3. Сохраните документ с именем Ex10.html в рабочей папке.

3.3 Задание 11

1. Создайте страницу с переадресацией на другой адрес (redirect).

2. Измените скрипт так, чтобы переадресация на другой адрес была с задержкой 5 секунд.

3. Сохраните документ с именем Ex11.html в рабочей папке.

3.4 Задание 12

1. Создайте HTML-документ, в котором будет 2 ссылки:

• первая ссылка должна ссылаться на PDF файл; при нажатии на нее выпадает сообщение с предупреждением о том, что для загрузки документа требуется программа Acrobat, и продолжить загрузку или нет; используйте для написания метод confirm(…) для подтверждения загрузки;

• вторая ссылка должна содержать такой код, чтобы при наведении на нее мыши менялся цвет фона документа на красный.

2. Сохраните документ с именем Ex12.html в рабой папке.

3.5 Задание 13

1. Создайте HTML-документ, содержащий любую картинку.

2. Добавьте скрипт с условиями:

• при наведении курсора мыши на картинку она увеличивается,

• при отведении курсора мыши – уменьшается до исходного размера.

Постройте скрипт через использование функций и событий MouseOver и MouseOut.

3. Сохраните документ с именем Ex13.html в рабочей папке.

3.6 Задание 14

1. Создайте HTML-страницу содержащую следующую форму заполнения данных:

Ваше имя: *

Пароль *

Подтверждение пароля*

Электронный адрес: *

Тема сообщения:

Сообщение:

Отправить Очистить

* - необходимые для заполнения поля

2. Добавьте скрипт, проверяющий следующие данные:

• заполнено ли поле имени,

• введен ли пароль и содержит ли он больше 4-х символов.

Используйте для этого свойство length данного поля,

• совпадают ли значения, введенные в оба поля для паролей,

• заполнено ли поле электронного адреса и содержит ли оно символ @,

• заполнено ли поле сообщения и содержит ли оно больше 10 символов,

3. При несоблюдении условий, курсор должен установиться в то поле, где пользователем введено неверное значение.

5. Сохраните документ с именем Ex15.html в рабочей папке.

Лабораторная работа №4. Объединение JavaScript и CSS.

4.1 Задание 15

1. Рассмотрите скрипт:

```
<head>
<title>h1</title>
<script language="JavaScript">
function colorchange()
{
head.style.color = "red";
}
</script>
</head>
<body>
<h1 id="head" onmouseover="colorchange()">Добро пожаловать на
нашу страницу!</h1>
</body>
</html>
```

2. Допишите скрипт страницы таким образом, чтобы красный цвет исчезал после отвода курсора мыши с заголовка.

3. Сохраните документ с именем Ex15.html в рабочей папке.

4.2. Задание 16

1. Рассмотрите скрипт:

```
<html>
<head>
<title>text decoration</title>
<script language="JavaScript">
function addunderline()
{
head.style.textDecoration = "underline";
}
function removeunderline()
{
head.style.textDecoration = "none";
}
```

```
</script>
</head>
<body>
<h1 id="head" onMouseover="addunderline()"
onMouseout="removeunderline()">
Добро пожаловать на нашу страницу!
</h1>
</body>
</html>
```

2. Допишите скрипт страницы таким образом, чтобы на одинарный щелчок мыши появлялось полоса над заголовком, а на двойной щелчок – текст зачеркивался. Используйте события onclick, ondblclick и значения рассматриваемого свойства overline и linethrough.

3. Сохраните документ с именем Ex18.html в рабочей папке.

4.3 Задание 17.

1. Создайте HTML-документ, содержащий любое изображение.

2. Поместите изображение в тег . Задайте для него абсолютное позиционирование со смещением вниз и влево на 500 пикселей.

1. Сохраните документ с именем Ex17.html в рабочей папке.

Лабораторная работа №5. Слои. Движущиеся элементы

5.1 Задание 18

1. Рассмотрите скрипт:

```
<html>
<head>
<title>simple animation</title>
<script language="JavaScript">
function moveTxt(){
if (anil.style.pixelLeft < 500){
60
anil.style.pixelLeft +=50;
setTimeout("moveTxt()", 5000);}}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anil" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

2. Измените скрипт страницы:

• добейтесь плавного передвижения текста;

• измените направление текста - задайте направление сверху вниз при помощи атрибута pixelTop.

3. Сохраните документ с именем Ex18.html в рабочей папке.

5.2 Задание 19

1. Рассмотрите скрипт:

```
<head>
<title>anima1</title>
<script language="JavaScript">
function moveTxt(){
```

```
if (anim.style.pixelTop <500){
anim.style.pixelTop +=2;
anim.style.pixelLeft +=2;
setTimeout("moveTxt()", 50);}}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anim" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

2. Измените направление текста. Задайте направление с верхнего правого угла экрана (приблизительно) по диагонали к середине экрана.

3. Сохраните документ с именем Ex19.html в рабочей папке.

5.3 Задание 20

1. Создайте HTML-страницу, на которой будет три слоя. Верхний и нижней представляют из себя статичные квадраты разного цвета с текстом, а меду ними должна проплывать любая картинка слева направо.

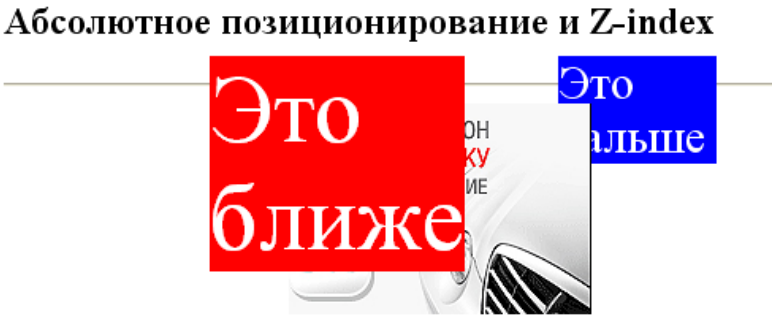
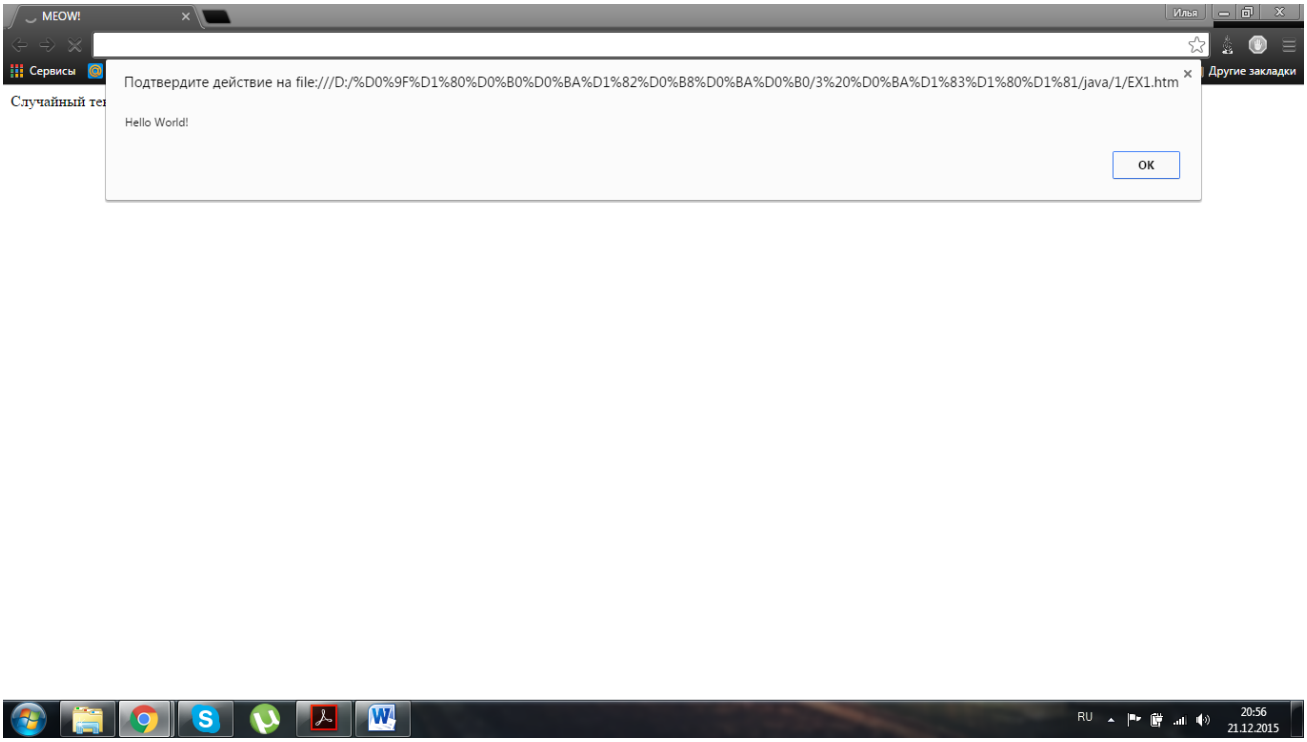


Рисунок 2 – Задание

2. Сохраните документ с именем Ex20.html в рабочей папке.

Пример выполнения практического задания

Вид окна

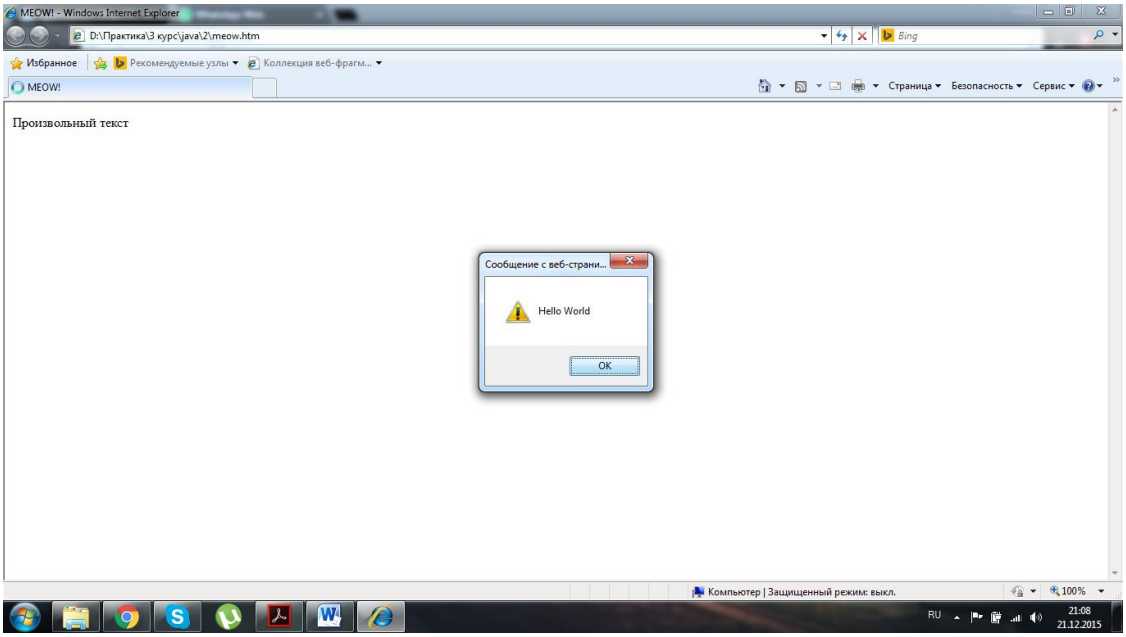


Вид скрипта

```
<HTML>
<HEAD>
<TITLE>
Ex1
</TITLE>
</HEAD>
<BODY>
<p1>
Случайный текст
</p1>
<br>
<script language="JavaScript">
alert("Hello World!")
</script>
<p2>
Случайный текст
</p2>
</BODY>
```

```
</HTML>
```

Вид окна




```
<HTML>
<HEAD>
<TITLE>
Ex2
</TITLE>
</HEAD>
<BODY>
<p1>
Произвольный текст
</p1><br>
<script language="JavaScript" src="meow.js">
</script>
<p2>
Произвольный текст
</p2>
</BODY>
</HTML>
```

Список литературы

1. Пол Вилтон, Джереми МакПик. JavaScript. Руководство программиста. СПб: Питер, 2009-720 с.
2. Стоян Стефанов. JavaScript. Шаблоны. СПб: Символ-плюс, 2011-272с.
3. Дунаев В. HTML, скрипты и стили. СПб: БХВ-Петербург, 2011- 816с.
4. Дронов В. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. СПб: БХВ-Петербург, 2011- 416 с.
5. Джон Поллок. JavaScript. Руководство разработчика. СПб: Питер, 2011-544 с.
6. Дэвид Макфарланд. JavaScript. Подробное руководство. Эксмо, 2009- 608 с.
7. Климов А. JavaScript на примерах. СПб: БХВ-Петербург, 2009-336 с.
8. Шафер С., HTML, XHTML и CSS. Библия пользователя. М.: Вильямс, 2010 – 656 с.
9. Лабберс К., Олберс Н., Салим К. HTML5 для профессионалов: мощные инструменты для разработки современных веб-приложений. М.: Вильямс, 2011 – 272 с.
10. Соболь Б.В. и др. Информатика. Феникс, 2009.
11. Кузнецова Л.В. Лекции по современным веб-технологиям. Москва, 2016.