Министерство науки и высшего образования Российской Федерации (Минобрнауки России)
Федеральное государственное образовательное бюджетное учреждение высшего образования
«Финансовый университет при Правительстве Российской Федерации»
(ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации»)

Санкт-Петербургский филиал

М.Ю. Лехмус

БАЗОВЫЕ ТЕХНОЛОГИИ ВЕБ-ПРОГРАММИРОВАНИЯ

Учебное пособие

КНОРУС
Москва
2024

УДК 004.42(075.8)
ББК 32.973.202-018.2я73
Л53

Утверждено планом изданий Санкт-Петербургского филиала Финуниверситета на 2024 год, протоколом заседания кафедры «Бизнес-информатика» Санкт-Петербургского филиала ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации» №10 от 5.12.2023, протоколом ученого совета Санкт-Петербургского филиала ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации» №07 от 19.12.2023.

Рецензенты:

М.Р. Вагизов, заведующий кафедры информационных систем и технологий, ФГБОУ ВО «Санкт-Петербургский государственный лесотехнический университет», канд. техн. наук, доц.,

Т.В. Рябикова, заведующий кафедры «Математика», Санкт-Петербургского государственного архитектурно-строительного университета, канд. физ.-мат. наук, доц.

Автор:

М.Ю. Лехмус, доцент кафедры «Бизнес-информатика» Санкт-Петербургского филиала ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации», канд. техн. наук

Лехмус, Михаил Юрьевич.

Л53 Базовые технологии веб-программирования : учебное пособие / М.Ю. Лехмус. — Москва : КНОРУС, 2024. — 88 с.

ISBN 978-5-406-13372-9

Учебное пособие подготовлено для студентов специальности «Бизнес-информатика» и служит овладению навыками и приобретением знаний по базовым технологиям, использующим языки веб-разработки контента и управления им. Рассмотрены основные приемы веб-разработки, как в объектно-ориентированной среде, так и в алгоритмическом контексте.

Подходит для студентов направления подготовки 38.03.01 «Экономика», 38.03.02 «Менеджмент», 38.03.05 «Бизнес-информатика» при изучении вопросов веб-разработки.

***Ключевые слова**: JavaScript; веб разработка; программный код; Java; объектно-ориентированная среда разработки.*

УДК 004.42(075.8)
ББК 32.973.202-018.2я73

ISBN 978-5-406-13372-9

Оглавление

ВВЕДЕНИЕ

Современные тренды веб-разработки диктуют необходимость изучения все большего количества новых информационных ресурсов и инструментов разработки, хотя базовые технологии по-прежнему являются востребованными.

Универсальный независимый язык программирования Java и его модификация JavaScript широко используются в разработке деск-топных программ, апплетов, мобильных приложений, разнообразного сетевого и веб-ориентированного программного обеспечения. Они имеют грандиозные возможности для реализации инновационных проектов и масштабных бизнес-решений. Только эксперты и опытные специалисты, следящие за последними тенденциями в мире Java, могут вполне использовать весь арсенал инструментов для быстрого решения поставленных задач.

В пособии излагаются базовые понятия и методы алгоритмического и объектно-ориентированного программирования.

Рассматривается методика установки комплекта инструментальных средств разработки, описывается применение инструментальных средства.

1. БАЗОВЫЕ ТЕХНОЛОГИИ WEB-РАЗРАБОТКИ В СРЕДЕ JAVASCRIPT

JavaScript – это объектно-ориентированный язык программирования, язык сценариев, с помощью которого возможно создавать интерактивные html-документы, производить вычисления, выполнять проверку допустимости данных без обращения к серверу, то есть с помощью него страницы сайтов становятся более интерактивными. Скрипты, то есть программы на JavaScript предоставляют возможность пользователю взаимодействовать с сайтами: заполнять формы обратной связи, оставлять комментарии, просматривать всплывающие подсказки и демонстрировать некую динамичность (например, смена картинок в блоке или плавно выпадающее меню).

Язык JavaScript сочетает в себе принципы и особенности объектно-ориентированного, функционального, событийно-ориентированного, декларативного и императивного программирования. Объектная ориентированность языка означает доступность использования программистами различных объектов, таких как объекты самого языка, являющиеся для него внутренними, так и объекты браузера и загруженного в этот браузер html-документы, которые являются внешними.

Сценарий JavaScript встраивается непосредственно в исходный текст html-документа и интерпретируется браузером по мере загрузки этого документа.

Для запуска написанных на языке JavaScript скриптов, вам необходим браузер, способный работать с JavaScript – например, Google Chrome или Microsoft Internet Explorer, а также текстовый редактор: Блокнот или Notepad.

Для работы создайте папку с именем: «И – ***, Иванов», где будет указан номер вашей группы и ваша фамилия.

JavaScript сценарии бывают двух типов: встроенными, т.е. их содержимое является частью html–документа, и внешними, содержимое JavaScript сценариев находится в другом файле с расширением .js. JavaScript–коды возможно внедрить в html-документы различными способами. Эти способы описаны ниже.

1.1. Размещение сценария JavaScript на странице html-документа

Для начала составим JavaScript сценарий, который вставит слова "Привет, мир!" непосредственно в html–документ.

Документ составляется по правилам языка html:

Код скрипта JavaScript встраивается в html-документ с помощью тега <script>.

Команда document.write() является одной из наиболее важных команд, используемых при программировании на языке JavaScript и позволяет записывать в поток документа строки текста. Этот метод используется, в том случае, когда требуется написать какой-либо текст в текущем документе.

То есть для объекта с именем document вызывается метод write.

Задание 1. Размещение JavaScript–кода на html–странице, используя элемент <script>

Запустите блокнот.

Введите текст:

```
<html>
<body>
<script language="JavaScript">
 <!--
 document.write("Привет, мир!");
 // -->
 </script>
 </body>
</html>
```

Сохраните, созданный вами документ в формате .html

Откройте html-страницу с помощью браузера.

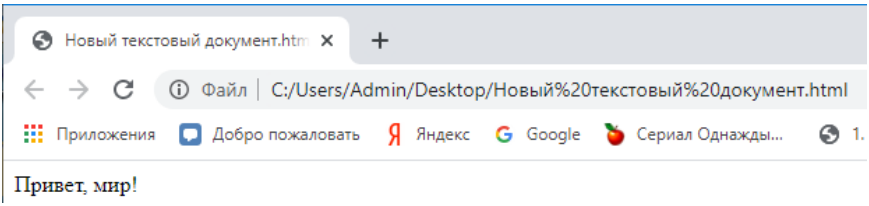

Рисунок 1.1. Результат размещения операторов языка JavaScript на странице

Задание 2. Размещения JavaScript–кода на html–странице с подгружаемым исходным текстом в виде гиперссылки

Для этого нужно разместить JavaScript код в отдельном файле и включить ссылку на этот файл на странице html документа. Браузер загружает оформленные подобным образом html–документы, а также загружает оформленные в отдельных файлах сценарии, и после подставляет их вместо соответствующих ссылок. Этот способ включения сценариев удобен, при использовании одного сценария в различных других документах html, или если сценарий имеет большой размер, а также если возникает необходимость скрыть исходный код от пользователей.

Ссылки на файлы с подгружаемым исходным текстом в виде гиперссылки оформляются с помощью параметра SRC тега <script>, который допускает указание URL адреса файла сценария.

Рассмотрим на примере использование параметра SRC. В исходном тексте документа html, имеется ссылка на файл сценария hello.js. Ссылка расположена между тегами <script> и </ script>, но между тегами нет строчек исходного кода. Текст исходного кода перенесен в файл hello.js

Создайте файл hello.js

Введите в нем текст:

```
document.write("<hr>");
document.write("Hello, world!");
document.write("<hr>")
```

3. Сохраните файл hello.js
4. Запустите блокнот.
5. Введите текст:

```
<html>
 <head>
 <title>Привет, мир!</title>
 </head>
 <body>
<script language="JavaScript" SRC="hello.js">
 </script>
 </body>
</html>
```

Сохраните, созданный вами документ в формате .html

Откройте html-страницу с помощью браузера.

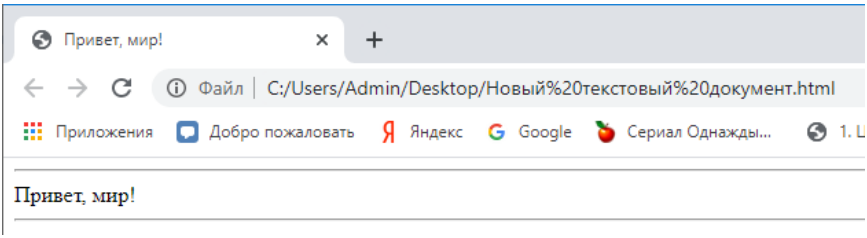

Рисунок 1.2. Результат размещения операторов языка JavaScript на странице с подгружаемым исходным текстом

Результат работы сценария, практически аналогичен результату работы выполненной в ходе задания 1 (рисунок 1.2), за исключением горизонтальных линий, которые выделяют приветственную фразу “Привет, мир!”. За создание этих линий отвечают использованные нами теги <hr>, которые мы вставили в тело документа html сценарием JavaScript.

Задание 3. Размещения JavaScript–кода на html–странице с использованием переменной и функций

Можно отметить активное применение функций и переменных в сценариях JavaScript. Переменные и функции должны быть определены и оформлены, с применением тегов <script> и </ script > а также должны быть расположены в области: заголовка документа, выделенной тегами <head> и </ head >.

```
< script language="JavaScript">
<!--
 var s = "Привет, мир!";
 function printHello()
 {
 document.write(s);
 }
// -->
</script>
```

В данном случае переменная s определяется при помощи оператора var, имеет имя s, и присвоенное начальное значение – текстовую строку "Привет, мир!".

Также в области заголовка html– документа должна быть определена функция с помощью ключевого слова function и иметь имя printHello. Вызов этой функции осуществляется из JavaScript сценария, расположенного в теле документа. Функция выводит в html– документ значение заранее определённой нами переменной s.

```
<script language ="JavaScript">
<!--
 printHello();
// -->
</script>
```

1. Запустите блокнот.
2. Введите текст:

```
<html>
 <head>
 <script language="JavaScript">
 <!--
 var s = "Привет, мир!";
 function printHello()
 {
 document.write(s);
 }
 // -->
 </script>
</head>
 <body>
<script language="JavaScript">
 <!--
 printHello();
 // -->
 </script>
 </body>
</html>
```

3. Сохраните, созданный вами документ в формате .html
4. Откройте html-страницу с помощью браузера.

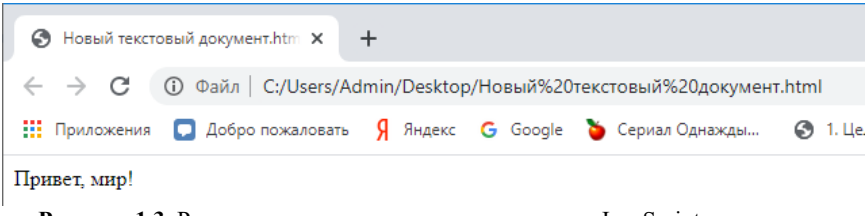

Рисунок 1.3. Результат размещения операторов языка JavaScript на странице с использованием переменной и функции

Задание 4. Размещения JavaScript–кода на html–странице с использованием диалогового окна

В языке JavaScript предусмотрены встроенные средства для отображения простейших диалоговых панелей, например, панели сообщений. Для использования диалоговой панели необходимо вызвать функцию alert. Функция alert() предназначена для вывода на экран пользователя предупреждающего диалогового окна с указанным сообщением и кнопкой "ОК". Она может использоваться для того чтобы донести до пользователя важную информацию.

Функция alert() должна иметь один обязательный параметр – текст сообщения, которое отображается в диалоговом окне.

```
<script language="JavaScript">
 <!--
 function printHello()
 {
 alert("Hello, world!");
 }
 // -->
 </script>
```

1. Запустите блокнот.
2. Введите текст:

```
<html>
 <head>
 <title>Привет, мир!</ title >
 <script language="JavaScript">
 <!--
 function printHello()
 {
 alert("Привет, мир!");
 }
 // -->
 </script>
 </head>
 <body>
JavaScript Test
<br>
 <script language="JavaScript">
 <!--
 printHello();
```

```
// -->
</script>
</body>
</html>
```

3. Сохраните, созданный вами документ в формате .html

4. Откройте html-страницу с помощью браузера.

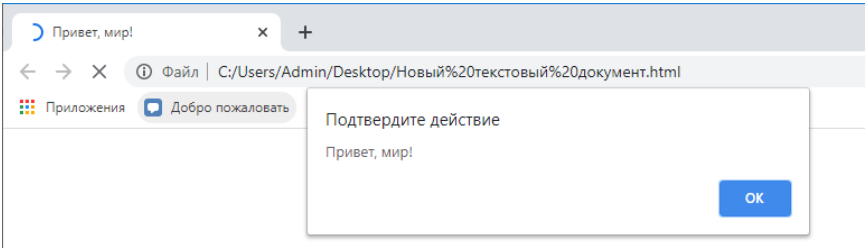

Рисунок 1.4. Вывод диалоговой панели

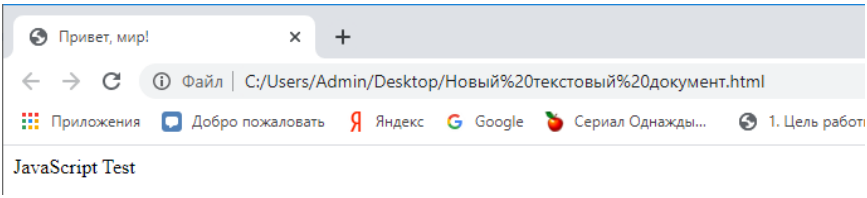

Рисунок 1.5. Результат размещения операторов языка JavaScript на странице с использованием диалоговой панели сообщений

Кроме представленного диалогового окна сценарии JavaScript также могут выводить на экран и более сложные диалоговые панели. Например, предоставлять возможность пользователю делать выбор из двух альтернатив или позволять пользователю вводить какую-либо информацию.

Задание 5. Размещения JavaScript–кода на html–странице с использованием обработчика событий

Рассмотрим использование средств обработки событий, которые присутствуют в языке JavaScript. Например, нам необходимо обработать такое событие, что, когда пользователь размещает курсор мыши на ссылке "Select me!" и тем самым пытается выбрать эту ссылку, на экране в этот момент должна появиться диалоговая панель с сообщением "Привет, мир!".

Для этого необходимо использовать тег <a>. Применение этого тега необходимо для осуществления организации ссылок на файлы различных объектов или другие документы html. В нашем примере поле ссылки параметра href (href="") пустое, но мы включим в тег <a> дополнительную конструкцию: onMouseover="alert(' Привет, мир!!');", которая указывает, что по прошествии события onMouseover (наведение мыши на объект) должна будет выполниться следующая строка программы JavaScript:

alert('Привет, мир!');

то есть должна отобразиться диалоговая панель.

Существуют также и другие html–события, которые также можно использовать, и при обнаружении которых JavaScript выполнит необходимый код.

Таблица 1.1

Используемые html события

Событие	Описание
onchange	HTML элемент был изменен
onclick	Пользователь кликнул мышкой на HTML элемент
onmouseover	Пользователь навел мышку на HTML элемент
onmouseout	Пользователь вывел мышку за пределы HTML элемента
onkeydown	Пользователь нажал на клавишу клавиатуры
onload	Браузер закончил загружать страницу

1. Запустите блокнот.
2. Введите текст:

```
<html>
 <head>
 <title>Hello world!</title>
 </head>
 <body>
 <h1>JavaScript Test</h1>
 <a href="" onMouseover="alert('Привет, мир!');">
Select me!</a>
 </body>
```

</html>

3. Сохраните, созданный вами документ в формате .html
4. Откройте html-страницу с помощью браузера.

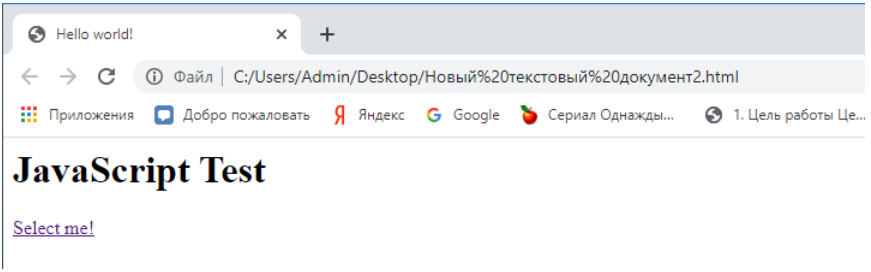

Рисунок 1.6. Результат запуска страницы документа с обработкой событий

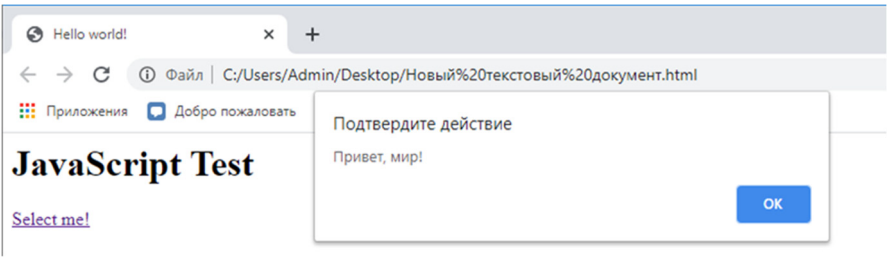

Рисунок 1.7. Результат при наведении мыши на объект

1.2. Переменные в JavaScript

Переменная – это именованный участок памяти, в который можно как сохранить некоторую информацию, так и получить её из неё. Переменная представляет собой идентификатор, которому присвоено некое значение. Переменные в JavaScript – это контейнеры для хранения различных данных.

В сценариях JavaScript использование переменных возможно, при адресации к ним по имени. Переменные в JavaScript делятся на два типа: глобальные и локальные. Доступ к глобальным переменным возможен из любого места сценария, локальные же переменные действуют только в пределах той функции областью которой ограничены и внутри которой эти переменные объявлены. использование переменных без их предварительного объявления при создании сценариев JavaScript возможно,

но исключением являются локальные переменные, уже определенные в функциях.

Все же рекомендуется объявлять переменные и присваивать им начальные значения перед их использованием. Это упростит отладку сценариев и уменьшит вероятность появления ошибок при составлении исходного текста, особенно если применяются и глобальные, и локальные переменные в коде одновременно.

Объявление переменных.

Объявление переменных в JavaScript осуществляется с помощью ключевого слова var, как это показано ниже:

var s;

var message;

Тип переменной обычно определяется в момент первого присвоения значения.

При выборе имени переменной необходимо следовать следующим правилам:

1. Имя переменной должно начинаться с буквы или с символов "_", "$" и должно состоять только из букв, цифр, а также символов "_", "$";

2. Имена переменных чувствительны к регистру, то есть регистр букв имени переменной имеет значение. Например, var message и var Message – разные переменные.

3. Имя переменной не должно совпадать с зарезервированными ключевыми словами JavaScript. Набор ключевых слов, определенный стандартом ECMA-262 представлен в таблице ниже, также в таблице содержится набор зарезервированных слов, которые не являются частью зыка в настоящее время но могут войти в его состав в будущих версиях языка:

Ключевые слова					
break	case	catch	class	const	continue
debugger	default	delete	do	else	enum
export	extends	false	finally	for	f unction
if	import	in	new	null	return
super	switch	this	throw	true	try
typeof	var	void	while	with	
Зарезервированные слова					
class	const	enum	export	extends	import
implements	let	private	public	yield	implements
interface	package	protected	static		

Также следует избегать использования имени встроенных объектов JavaScript, свойств и методов:

Array	Date	eval	function
hasOwnProperty	Infinity	isFinite	isNaN
isPrototypeOf	length	Math	NaN
name	Number	Object	prototype
String	toString	undefined	valueOf

1.3. Присвоение значения переменным

При объявлении тип переменной не указывается. Этот тип присваивается переменной только тогда, когда ей присваивается какое-либо значение. Присвоить значение переменной можно сразу же при ее создании. Для присвоения переменной значения необходимо использовать оператор «=».

Типы данных в JavaScript

В языке JavaScript существует несколько типов данных.

number	(число)
string	(текстовая строка)
boolean	(логическое значение)
null	(специальное значение null)
undefined	(специальное значение undefined)
symbol	(символ, используется в особых случаях)

Например, объявим переменную s с помощью ключевого слова var, затем запишем в нее текстовую, тогда тип переменной станет строковым:

```
var s;
s = "Привет, мир!";
```

В языке JavaScript возможна сокращенная запись объявления переменной и присвоения значения. Рассмотренный выше пример в таком случае будет иметь следующий вид:

```
var s = " Привет, мир!";
```

Присвоение переменной численного значения возможно в любом месте программы:

s = 4;

При этом измениться тип переменной, причем браузер не отобразит никаких предупреждающих сообщений в процессе интерпретации сценария, так как такое изменение является допустимым.

Рассмотрим различные типы данных в JavaScript.

Число

Этот тип используется для числовых значений. Числа в языке JavaScript бывают двух типов: целые числа (integer) и числа с плавающей точкой (floating-point number). Язык JavaScript допускает использование чисел в различных системах счисления: в десятичной, восьмеричной, шестнадцатеричной. Числа с плавающей точкой представляют собой числа с дробной десятичной частью, либо это числа, выраженные в экспоненциальном виде.

25	В десятичной системе значения числовых переменных
0137	В восьмеричном формате
0xFF	Целое число в шестнадцатеричном формате
386.7	Число с плавающей десятичной точкой
25e5	Число в научной нотации, равно 2500000

Тип null

Переменная может иметь специальное значение null которое используется для представления несуществующих объектов и указывает на отсутствие значения:

s = null;

Такое присвоение применяется в тех случаях, когда нужно объявить и проинициализировать переменную, не присваивая ей никакого типа и начального значения.

Строковый тип

Текстовая строка – это последовательность символов, заключенных в одинарные или двойные кавычки. Этот тип используется для хранения строки символов. Пустой набор символов, заключенный в одинарные или двойные кавычки, является пустой строкой. Число, заключенное в кавычки, также является строкой.

"Привет, мир!"

"" , причем эта строка – пустая

"12345"

'Это текстовая строка'

Отметим следующие присвоения.

szStr=""

szStr1=null

Отмеченные выше две строки не эквивалентны, так как в первой переменной szStr хранится текстовая строка, но она пустая, во второй переменной ничего не хранится так как ей присвоено значение null.

Логический тип данных

Логические данные имеют только два значения: true (истина), и false (ложь) и предназначены для выполнения операций сравнения (например, =, ==, <, >) и для использования в условных операторах (наподобие if и while), помогая управлять ходом выполнения программы.

Данные неопределенного типа

Переменная имеет неопределенный тип в том случае если она была объявлена, но значение ей еще ни разу не было присвоено. Например, мы имеем в строке сценария объявленную переменную MyVariable, имеющую неопределенный тип:

var MyVariable;

Выражения JavaScript

Выражения в JavaScript представляют собой комбинации операндов и операторов, в результате вычисления которой получается одно единственное значение.

Операнды – это данные, обрабатываемые сценарием JavaScript. В качестве операндов могут быть как простые типы данных, так и сложные, а также другие выражения.

Операторы – это символы языка, выполняющие различные операции с данными. Операторы могут записываться с помощью символов пунктуации или ключевых слов.

В JavaScript существует несколько типов операторов.

1. Оператор присваивания (=). Используется для присвоения переменным каких-либо значений.

var name2=10;

2. Арифметическое выражение

Арифметические операторы необходимы для выполнения математических операций, и работы с числовыми операндами (или переменными, хранящими числовые значения). Вычисляемым значением арифметического выражения является число. Создаются с помощью арифметических операторов.

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

3. Операторы инкремента и декремента

Операции инкремента и декремента производят увеличение и уменьшение значения операнда на единицу, чаще всего используются для увеличения счетчика в цикле. В качестве операнда может быть переменная, элемент массива, свойство объекта.

4. Операторы сравнения

Операторы сравнения необходимо использовать для сопоставления операндов. Результатом использования этих операторов должно быть одно из двух значений – true или false. Операндами являются не только числа, но и строки, логические значения и объекты.

Оператор/Операция	Описание
== Равенство	Проверяет две величины на совпадение, допуская преобразование типов. Возвращает true, если операнды совпадают, и false, если они различны.
!= Неравенство	Возвращает true, если операнды не равны
=== Идентичность	Проверяет два операнда на «идентичность», руководствуясь строгим определением совпадения. Возвращает true, если операнды равны без преобразования типов.
!== Неидентичность	Выполняет проверку идентичности. Возвращает true, если операнды не равны без преобразования типов.
> Больше	Возвращает true, если первый операнд больше второго, в противном случае возвращает false.
>= Больше или равно	Возвращает true, если первый операнд не меньше второго, в противном случае возвращает false.
< Меньше	Возвращает true, если первый операнд меньше второго, в противном случае возвращает false.
<= Меньше или равно	Возвращает true, если первый операнд не больше второго, в противном случае возвращает false.

5. Логические операторы

Логические операторы позволяют комбинировать условия, возвращающие логические величины. Чаще всего используются в условном выражении if.

Оператор/Операция	Описание
&& Логическое И	Возвращает true, только если оба операнда истинны. При выполнении операции сначала проверяется значение первого операнда. Если оно имеет значение false, то значение второго оператора не учитывается и результату выражения присваивается false.
|| Логическое ИЛИ	Возвращает true, если хотя бы один операнд истинен, т.е. проверяет истинность как минимум одного условия.
! Логическое НЕ	Изменяет значение оператора на обратное - с true на false и наоборот.

Задание 6. Использование выражений

Используя арифметические операторы необходимо выполнить математические операции. Арифметические операторы работают с числовыми операндами (или переменными, хранящими числовые значения), возвращая в качестве результата числовое значение.

Если один из операндов является строкой, интерпретатор JavaScript попытается преобразовать его в числовой тип, а после выполнить соответствующую операцию. Если преобразование типов окажется невозможным, будет получен результат NaN (не число).

1. Запустите блокнот.
2. Введите текст:

```
<html>
 <body>
 <h1> Пример арифметических операций </h1>
 <table>
 <script language="JavaScript">
 <!--
 var x = 5, y = 8, z, z1, z2, z3, z4, z5;
z = x + y + "<br>"; // вернет 13
z1 = x – y + "<br>"; // вернет -3
z2 = – y + "<br>"; // вернет -8
```

```
z3= x * y + "<br>"; // вернет 40
z4 = x / y + "<br>"; // вернет 0.625
z5 = y % x + "<br>"; // вернет 3
document.write(z+z1+z2+z3+z4+z5);
 // -->
 </script>
 </table>
 </body>
</html>
```

3. Сохраните, созданный вами документ в формате .html
4. Откройте html-страницу с помощью браузера.

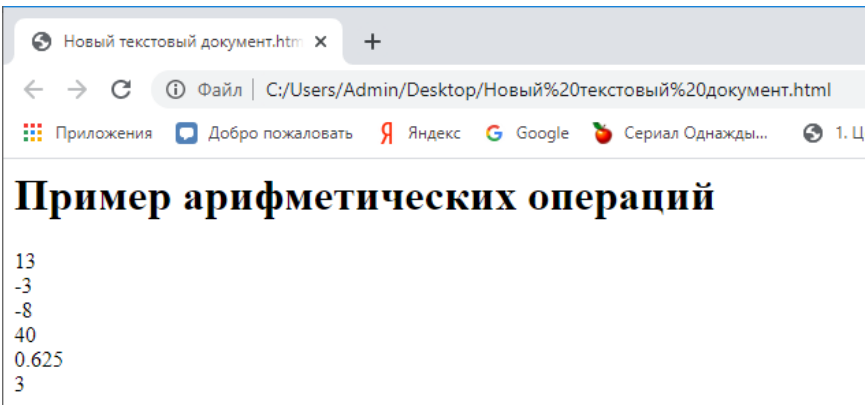

Рисунок 1.8. Результат использования арифметических операторов

Преобразование типов данных

Если в выражениях встречаются переменные разных типов, интерпретатор JavaScript может автоматически преобразовывать численные данные в текстовые строки. Обратное же преобразование (строк в числа) приходится выполнять с помощью специальных функций, таких как parseInt и parseFloat. Поясним это, выполнив задание 8.

Задание 7. Преобразование типов данных

Допустим, мы имеем переменную szTextBuf объявим и проинициализируем ее пустой строкой.

```
var szTextBuf = "";
```

Далее присвоим этой строке сумму числа 4 и двух текстовых строк:

```
szTextBuf = 4 + " – число четыре" + " <br>";
```