**МИНОБРНАУКИ РОССИИ**
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационной безопасности

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_19_» ___04___ 2021 г.

# СОЗДАНИЕ САЙТОВ НА ЯЗЫКЕ JAVASCRIPT И ОБЕСПЕЧЕНИЕ ИХ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Методические указания по выполнению лабораторных работ
для студентов направления подготовки (специальности)
02.03.03 математическое обеспечение и администрирование
информационных систем

Курск 2021

2

УДК 004.56.5(076.5)
Составитель: А.Л. Ханис

Рецензент
Кандидат технических наук, доцент кафедры
информационной безопасности А.Г. Спеваков

**Создание сайтов на языке JavaScript и обеспечение их информационной безопасности** : методические указания по выполнению лабораторных работ студентов всех форм обучения / Юго-Зап. гос. ун-т; сост.: А.Л. Ханис. - Курск, 2021. - 41 с.: ил. 12, табл. 1. - Библиогр.: с. 41.

Содержат краткие теоретические положения о языке программирования JavaScript, а также методах защиты информации с помощью Java-скриптов, предназначенных для защиты контента от несанкционированного копирования.

Методические указания соответствуют требованиям программы по направлению подготовки бакалавров: математическое обеспечение и администрирование информационных систем.

Предназначены для студентов направления подготовки бакалавров 02.03.03.

Текст печатается в авторской редакции

Подписано в печать 19.04.21. Формат 60x84 1/16.
Усл.печ. л. 2,38 Уч.-изд. л. 2,15. Тираж 100 экз. Заказ. 419 Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

## Введение

В сознании большинства пользователей Интернет ассоциируется с тремя основными информационными технологиями:

- электронная почта (e-mail);
- файловые архивы FTP;
- World Wide Web.

Технология World Wide Web предоставляет простой интерфейс для доступа к разнообразным сетевым ресурсам. Это привлекло внимание коммерческих структур и привело к лавинообразному росту числа пользователей WWW.

Динамические сайты, в которых веб-страницы генерируются или формируются создаются динамически в процессе исполнения запроса пользователя. Динамические сайты бывают двух типов. В первом типе сайтов, веб-страницы генерируются или формируются из данных хранящихся на сервере в базе данных. Во втором типе сайтов веб-страницы генерируются на стороне клиентского приложения (в браузере).

Для управления клиентскими сценариями просмотра страниц Web-сайта можно использовать языки программирования этих сценариев, например, JavaScript. Клиентские сценарии широко применяются для решения таких задач, как проверка информации, введенной пользователем в формы, перед ее отправкой на сервер или для программирования различных ответных реакций на действия пользователя, делающих веб-страницы интерактивными.

В данном методическом пособии рассматриваются основы языка программирования JavaScript. Рассмотрены Java-скрипты для защиты контента от несанкционированного копирования.

## 1. Методы ввода и вывода информации в языке JavaScript. Краткие теоретические положения

Для ввода и вывода информации в языке JavaScript используются методы alert(), prompt(), confirm(). Эти методы генерируют различные окна сообщений. Метод alert() отображает окно предупреждения с соответствующим сообщением. Окно содержит кнопку Ok. Аргументом данного метода является строка.

Метод prompt () служит для получения данных от пользователя. При его вызове отображается окно приглашения с текстовым полем. Метод может содержать два аргумента. Оба эти аргумента должны быть строками. Первый аргумент – сообщение, которое отображается в окне. Второй аргумент – это текст по умолчанию, который должен появиться в соответствующем поле. Также этот метод возвращает значение, которое также является строкой – это текст, который ввел пользователь. Метод confirm() отображает окно подтверждения, которое сходно с окном предупреждения, генерируемое методом alert(), но содержит 2 кнопки: Ok и Cancel.

Также для вывода информации в языке JavaScript используются метод document.write ( ), который позволяет дописывать строку текста прямо в страницу.

**Задание №1**

**Цель работы:** изучить методику ввода и вывода информации в языке JavaScript.

**Назначение программы:** в данной программе выдается окно с запросом ―Здравствуйте, как Вас зовут?”. Пользователь вводит имя, например, Сергей. Программа выдает окно с сообщением: ―Рад Вас видеть, Сергей. Спасибо Вам, что зашли на мой сайт”.

**Текст программы:**

```
<html>
<head>
<title>сценарий 1</title>
<script>
var nameUser;
nameUser=prompt("Здравствуйте, как вас зовут?","аноним");
alert("Рад Вас видеть"+nameUser+"Спасибо Вам, что зашли на мой сайт");
</script>
</ head>
<body>
</ body>
</ html>
```

**Задание №2**

**Цель работы:** изучить методику применения метода document.write ( )

**Назначение программы:** программа 10 раз выводит сообщение ―Привет, мир!!!‖

**Текст программы:**

```
<html>
<head>
<title> Сценарий вывода информации </title>
</head>
<body>
<script type="text/javascript">
for (var i=0; i<10; i++)
{
document.write("Привет, мир!!!");
}
</script>
</body>
</html>
```

## 2. Переменные и массивы в языке JavaScript. Краткие теоретические положения

Рассмотрим примеры объявления и инициализации переменных в языке JavaScript:

```
var x,y,z;
var s=1.34e-5, msg=”Строка”, Flag=false;
```

Пока переменная не получит значение, она не определена – undefined. Чтобы определить тип переменной можно использовать операцию typeof, например, typeof(msg). Она возвращает возвращает строку с типом переменной, в данном случае - string. Следующий оператор выводит сообщение о значении переменной и типе переменной:

```
Document.write(―Переменная msg=”+msg+” – “+typeof(msg)
+”\n”);
```

Сообщение будет иметь вид:

**Переменная msg=Строка - string**

В языке JavaScript для использования нескольких данных одного типа используются массивы. Нумерация элементов массива

начинается с нуля. Язык JavaScript позволяет не перечислять элементы по одному, а сразу вывести все элементы массива. Для этого необходимо обратиться к самому массиву, в результате чего получится строка, в которой все элементы массива перечислены через запятую. Узнать длину массива можно с помощью свойства length. Например, следующая строка выведет размер массива stars:

alert(stars.length)

Пример объявления массива, задания значений элементов массива и вывода элементов массива рассмотрены в практическом занятии № 17.

**Здание №3**

**Цель работы:** изучить методику обработки массивов в языке JavaScript

**Назначение программы:** в программе реализован массив из 5 элементов (звезды), показан вывод первого элемента массива и всех элементов массива.

**Текст программы:**

```
<html>
<title>Инициализация массива</title>
<script>
//Объявляем массив и инициализируем значения элементов
var stars=new Array('Сириус','Канопус', 'Арктур', 'Капелла', 'Вега');
//Вывод первого элемента массива
alert(stars[0]);
//Вывод всех элементов массива
alert(stars);
</script>
</head>
<body>
</body>
</html>
```

**Задание №4**

**Цель работы:** разработать программу обработки массива в языке JavaScript

**Назначение программы:** программа должна находить сумму элементов массива, длину массива и среднее арифметическое значение элементов массива. Программа должна выводить результаты с помощью метода document.write() или метода alert().

## 3. Основные операторы в языке программирования JavaScript

### Краткие теоретические положения

Оператор if..else управляет механизмом ветвления, позволяет выбирать одно из альтернативных действий в зависимости от значения условия. Синтаксис данного оператора имеет вид:

```
if (логическое выражение)
{
операторы 1
}
else
{
операторы 2
}
```

Сначала вычисляется логическое выражение, затем если оно равно true, выполняются операторы 1, если оно равно false, то выполняются операторы 2. Альтернативная ветвь else может отсутствовать. В этом случае, если логическое выражение равно false, то управление передается следующему оператору и продолжается выполнение сценария.

Оператор switch позволяет сравнить значение с множеством других. Синтаксис данного оператора имеет вид:

```
switch (выражение)
{
case значение1:
операторы
case значение 2:
операторы
…….
default:
операторы
}
```

В данном операторе определяется значение выражения. Значение, возвращаемое выражением, сравнивается со значениями, указанными после ключевых слов case. Если имеет место совпадение, выполняется соответствующий блок операторов. Если не найдено ни одно совпадение, то выполняются операторы после инструкции default.

Оператор for служит для создания цикла. Он имеет следующий синтаксис:

for (выражение инициализации; выражение условия; выражение цикла)

{

операторы

}

Выражение инициализации служит для задания начального значения цикла. Выражение условия позволяет прекратить цикл, когда условие перестанет выполняться, то есть примет значение false. Выражение цикла осуществляет инкремент или декремент счетчика цикла.

Оператор while сходен с оператором for, но он не производит инициализацию и инкремент счетчика в своем объявлении. Синтаксис этого оператора следующий:

while (выражение условия)

{

Операторы

}

Оператор do..while практически идентичен оператору while, но, поскольку в нем проверка условия осуществляется в конце, он гарантирует выполнение операторов по крайней мере один раз:

do

{

Операторы

} while (выражение условие)

**Задание №5**

**Цель работы:** изучить методику применения оператора switch

**Назначение программы:** программа осуществляет перевод названия животных с английского языка на русский.

**Текст программы:**

```
<html>
<head>
<title>переводчик</title>
<script>
var trans;
var beast=prompt ("Введите название животного","dog");
switch (beast)
{
case "dog":
trans="собака";
break;
case "cat":
trans="кошка";
break;
case "cow":
trans="корова";
break;
default:
trans="неизвестное животное"
}
alert(beast+"---"+trans);
</script>
</ head>
<body>
</ body>
</ html>
```

**Задание №6**

**Цель работы:** изучить методику применения оператора while.

**Назначение программы:** программа осуществляет расчет факториала числа.

**Текст программы:**

```
<html>
<head>
<title> факториал с помощью оператора while </title>
<script>
```

```
var f=1;
var x=prompt("введите число","5");
x=+x;
var i=1;
while(i<=x)
{
f=f*i;
i++;
}
alert(f);
</script>
</head>
<body>
</body>
</html>
```

**Задание №7**

**Цель работы:** изучить методику применения оператора if..else, программу разработайте самостоятельно.

**Назначение программы:** программа рассчитывает значение функции:

x+10, при x<0

F(x) = $x^2$ +4, при 0 x 5

5/x, при x>5

## 4. Функции в языке программирования JavaScript. Краткие теоретические положения

Функция – это набор команд, объединенных под общим именем для выполнения определенной задачи. Синтаксис функции имеет следующий вид:

```
function ИмяФункции ([аргумент1] [,..аргумент N])
{
операторы
}
```

Ключевое слово function объявляет функцию с именем ИмяФункции. Функциям могут передаваться аргументы, которые перечисляются через запятую в скобках рядом с именем функции. В качестве аргументов могут выступать переменные, значения и

выражения. Функции можно объявить в любом месте внутри элемента SCRIPT. Тем не менее, рекомендуется объявить функции в блоке HEAD до начала выполнения основного кода, иначе функция может быть недоступной в момент ее вызова.

Функции могут возвращать значение. Для этого необходимо использовать оператор возврата return. Функции можно использовать в выражениях. Функция может объявляться в выражении. Обычно в этом случае функция присваивается переменной, которая затем может использоваться в выражении:

```
var cube =function(x) {return x*x*x}
alert(cube(2)+cube(3));
```

В данном случае объявляется переменная-функция, вычисляющая куб числа.

**Задание №8**

**Цель работы:** изучить методику использования функций

**Назначение программы:** программа рассчитывает в зависимости от выбранного режима сумму или произведение элементов массива. Для расчета суммы и произведения использованы две функции sum() и product().

**Текст программы:**

```
<html>
<head>
<title>Пример использования функций</title>
<script>
//Глобальный массив
var a= new Array (1,2,4,8);
function sum()
{
var s=0; //локальная переменная для хранения суммы
//Цикл по всем элементам массива a
for (var i in a)
{
s=s+a[i];
}
alert("Сумма элементов массива="+s);
}
```

```
function product()
{
var p=1; //локальная переменная для хранения произведения
for (var i in a)
{
p=p*a[i];
}
alert("Произведение элементов массива="+p);
}
</script>
</head>
<body>
<script>
var x=prompt("Найти сумму или произведение (0 или 1)?","0");
if (x=="0")
{
sum();
}
if (x=="1")
{
product();
}
</script>
</body>
</html>
```

**Задание №9**

**Цель работы:** изучить методику использования значения функции

**Назначение программы:** функция sum(a,b) возвращает сумму a и b, затем значение функции используется в выражении и результат выражения выводится на экран методом alert:

**Текст программы:**

```
<html>
<head><title>Использование значения функции</title>
<script>
function sum(a,b)
```

```
{
return (a+b);
}
</script>
</head>
<body>
<script>
var x1=5, x2=6, x3=3, x4=3;
//Отображаем (x1+x2)*(x3+x4)
alert (sum(x1,x2)*sum(x3,x4));
</script>
</body>
</html>
```

**Задание №10**

**Цель работы:** получить навыки самостоятельной разработки программ с использованием функций на языке JavaScript

**Назначение программы:** программа рассчитывает число сочетаний без повторений из $n$ элементов по $m$ элементов по формуле:

$$C_n^m = \frac{n!}{m!(n-m)!}.$$

При решении задачи расчет факториала надо выполнить с помощью специальной функции.

## 5. Обработчики событий в языке программирования JavaScript. Краткие теоретические положения

В языке JavaScript часто используются реагирования на события, которые могут быть вызваны пользователем или браузером. К событиям относятся открытие новой страницы, перемещение указателя мыши, щелчок кнопкой мыши. Каждое событие имеет соответствующий обработчик, который автоматически реагирует на возникшее событие. Например, если посетитель веб-страницы нажимает некоторую кнопку, происходит событие Click и вызывается обработчик событий onClick. Если указатель мыши пересекает гипертекстовую ссылку, происходит событие MouseOver, которому соответствует обработчик событий onMouseOver. Рассмотрим сценарии с данными обработчиками событий.

**Задание №10**

**Цель работы:** изучить методику использования обработчиков событий onMouseOver и onMouseOut для динамической смены изображений-ссылок на веb-странице

**Назначение программы:** изначально на веб-странице отображается рисунок a.jpg. Когда посетитель наводит указатель мыши на изображение, картинка меняется на b.jpg. Это результат действия обработчика onMouseOver. Когда пользователь убирает указатель с картинки, снова появляется изображение a.jpg. Здесь действует команда onMouseOut.

**Текст программы:**

```
<html>
<head>
<title>JavaScript</title>
</head>
<body topmargin=70>
<center>
<h1> Демонстрация обработчика события наведения указателя мыши </h1>
<br>
<a href="primer1.html" onMouseOver="document.pic.src='b.jpg'" onMouseOut="document.pic.src='a.jpg'"> <img SRC="a.jpg" name="pic" hspace=40 border=2></a>
</center>
</body>
</html>
```

**Задание №11**

**Цель работы:** изучить методику использования обработчика событий onClick

**Назначение программы:** программа по ссылке переходит на поисковую систему Rambler. Но предварительно по обработчику события onClick выдается сообщение «Желаем удачного поиска информации»

**Текст программы:**

```
<html>
```

```
<head>
<title>JavaScript</title>
</head>
<body topmargin=70>
<center>
<h1> Демонстрация обработчика щелчка кнопки мыши </h1>
<br>
<a href="http://www.rambler.ru" onClick="alert('желаем удачного по-иска информации');"> Поиск </a>
</center>
</body>
</html>
```

**6. Окна в языке программирования JavaScript. Краткие теоретические положения**

Объект window представляет окно браузера или фрейм. С помощью сценария можно создавать или удалять окна, загружать в них документы и записывать данные. Окно создается методом open () объекта window.

Window.open (URL-адрес, имя, опции, замена);

URL - адрес – строка с URL – адресом документа, который следует загрузить в открываемое окно; если вы не хотите загружать в окно никакой документ, то укажите пустую строку — “.

Имя - строка с именем окна, которое позже может стать значением атрибута target в тегах <a> ссылок, если вы не будете ссылаться на данное окно, то укажите пустую строку — “. Если вам нужно, чтобы страница загружалась в маленьком окне, то данный параметр должен иметь значение joe.

Опции – строка, в которой через запятую указаны дополнительные параметры оформления окна.

Замена – необязательное логическое значение, указывающие, должен ли документ по заданному URL – адресу заменить содержимое окна (применительно к уже существующим окнам).

Пример открытия окна:

mywindow= open(http://www.yandex.ru, —Yandex”, —width=400, height=400”)

Ддя закрытия окна служит метод close(). Однако выражение window.close() закрывает главное окно, а не то окно, которое создано методом open(). В этом случае необходима ссылка на созданное окно, которую следует сохранить в глобальной переменной.

Пример закрытия окна:

Var objwin=window.open(―mypage.htm”, ―Моя страница”)

Objwin.close()

При создании окна методом open() с помощью третьего параметра можно указать в строке множество дополнительных параметров, разделенных запятыми.

Наиболее популярными являются следующие параметры:

width, height – ширина и высота окна в пикслях;

left, top – горизонтальная и вертикальная координаты относительно левого верхнего угла экрана в пикселях;

menubar – разрешение/запрет отображения панели меню;

toolbar - разрешение/запрет отображения панели инструментов;

scrollbars – разрешение/запрет отображения полос прокрутки;

resizable – разрешение/запрет изменения размеров окна пользователем;

location – параметр определяет, будет ли отображаться поле ввода адреса HTML – документа;

status – отображение строки состояния.

Логические значения для данных параметров указывают как yes и no.

Также возможен следующий вариант синтаксис команды открытия окна:

Window.open(_адрес URL‘, ‘имя окна‘, config=‘параметр 1, параметр 2, параметр n‘)

Параметр config показывает, что следующие атрибуты относятся к конфигурации нового окна и определяют его внешний вид. Все атрибуты параметра config перечисляются через запятую без пробелов и заключаются в одинарные кавычки.

У объекта window имеется синоним self, используемый при обращении к окну, содержащему текущий элемент.

**Задание №12**

**Цель работы:** изучить методику создания и использования окон в языке JavaScript

**Назначение программы:** в маленькое окно загружается фотография (файл foto.jpg). По нажатию кнопки «**Увеличить**» создается новое окно большого размера и в него загружается данная фотография.

**Текст программы:** программа реализована в виде двух файлов. Имя второго файла – foto2.html.

```
<html>
<head>
<title>JavaScript</title>
<script language="JavaScript">
function openwin()
{
window.open('foto2.html','joe', config='height=300, width=400, toolbar=no, menubar=no, scrollbars=no, resizable=no, loca-tion=no, status=no')
self.name="main window"
}
</script>
</head>
<body>
<h3>Фотография</h3>
<img src="foto.jpg" border="0" width="120" height="90" alt="Фотография"> <br> <br>
<form>
<button type="SUBMIT" onClick="openwin()">Увеличить </button>
</form>
</body>
</html>
```

**Файл foto2.html**

```
<html>
<head>
<title> Фотография </title>
</head>
<body background="foto.jpg">
```

```
</body>
</html>
```

**Задание №13**

**Цель работы:** получить навыки самостоятельной работы создания фотогалереи в языке JavaScript.

**Назначение программы:** на веб-странице имеются несколько фотографий маленького размера. При щелчке мышью на фотографии увеличенная фотография должна открываться или в отдельном фрейме или в отдельном окне. Добавьте кнопки перехода к следующей и предыдущей фотографии.

## 7. Анимация изображения в языке программирования JavaScript. Краткие теоретические положения

Для анимации изображения необходимо через равные промежутки времени изменять атрибут src в изображении. Для создания анимации используют одну из функций объекта Window: setTimeout () или setInterval (). Рассмотрим пример анимации изображения.

**Задание №14**

**Цель работы:** изучить методику использования анимации изображений в языке JavaScript.

**Назначение программы:** В данном сценарии последовательно происходит анимация изображений img5.jpg, img4.jpg, img3.jpg, img2.jpg, img1.jpg, img0.jpg. Это происходит с помощью функции animation (). Данная функция получает управление после щелчка кнопкой мыши на рисунке. После чего функция изменяет рисунок, уменьшает счетчик и порождает другой свой экземпляр, выполнение которого откладывается на одну секунду. Когда счетчик становится отрицательным, процесс анимации прекращается.

**Текст программы:**

```
<html>
<head>
<meta charset="utf-8">
<title>Анимация изображения</title>
<script>
```

```
//Создаѐм массив для хранения изображений
numerals=new Array(6);
//В цикле создаѐм объекты изображений и загружаем рисунки
for(var i=1;i<6;i++){
//Создаѐм объекты изображений
numerals[i]=new Image();
// Загружаем все рисунки
numerals[i].src="img"+i+".jpg";
}
//Функция, осуществляющая анимацию
function animation(count){
//Загружаем текущий рисунок
document.images[0].src="img"+count+".jpg";
count--; //Уменьшаем счѐтчик
//Пока есть изображения, рекурсивно вызываем функцию animation()
if (count>=0){
//Открываем вызов функции на секунду
setTimeout("animation("+count+");",1000);
}
}
</script>
</head>
<body>
<h2 style="text-align : center;">Анимация изображения</h2>
<p style="text-align : center;" >
<img src="img5.jpg" width="300" height="300" onClick="animation(5)"> </p>
</body>
</html>
```

**8. Создание меню в языке программирования JavaScript. Краткие теоретические положения**

Одна из главных задач, которую необходимо решать программисту при разработке сайта, это создание удобной системы навигации (меню). Для этого можно использовать обычные гиперссылки, а можно воспользоваться средствами языка JavaScript.

Рассмотрим пример создания простого горизонтального меню. Для создания такого меню использовался обычный маркированный список со ссылками внутри, а также каскадные таблицы стилей: стиль ul li позволяет выстроить список в линию и убрать маркер; стиль li a описывает внутренние отступы и внешний вид меню. Переход по ссылкам в меню осуществляется при помощи обработчика событий onClick.

**Задание №18**

**Цель работы:** изучить методику создания меню в языке JavaScript.

**Назначение программы:** программа создает горизонтальное меню следующего вида:

О нас Новости Услуги Контакты

Рис 7. Горизонтальное меню.

**Текст программы:**

```
<html>
<head>
<title>Простое горизонтальное меню</title>
<!--Определяем стиль -->
<style type="text/css">
ul li{list-style-type:none;display:inline}
li a{font-family:Tahoma,Helvetica,sans-serif;text- decoration:none;color:#fff}
li a{display:block;float:left;padding:4px 10px;background:193D6A}
li a:hover{background:#408BE8;padding:7px 14px;position:relative;top:-3px}
</style>
</head>
<body>
<ul>
<li><a href="#" onClick="location.href='about.html';">
О нас</a></li>
<li><a href="#" onClick="location.href='news.html';">
Новости</a></li>
```

```
<li><a href="#" onClick="location.href='serv.html';">
Услуги</a></li>
<li><a href="#" onClick="location.href='cont.html';">
Контакты</a></li>
</ul>
</body>
</html>
```

**Дополнительные задания:**

1. Создайте веб-страницы, на которые указывают пункты меню и убедитесь в правильной работе сайта.

2. Измените цветовую раскраску меню следующим образом: фон меню – зеленый цвет, маркер, выбирающий пункт меню – красный цвет, текст – черный цвет.

**Задание №19**

**Цель работы:** получить навыки самостоятельной разработки меню в языке JavaScript.

**Назначение программы:** реализация вертикального раскрывающегося меню. В исходном состоянии меню выглядит подобно кнопке (прямоугольник с надписью). При наведении указателя мыши на данный прямоугольник меню раскрывается, показывая опции (пункты) своего подменю. При уходе указателя мыши с прямоугольника раскрывшегося меню последнее сворачивается и принимает исходное состояние. Внешний вид меню представлен на рис. 8. При щелчке мышью на пункте меню загружается соответствующая поисковая система.


Рис 8. Вертикальное раскрывающиеся меню.

## 9. Защита информации с помощью аутентификации в языке программирования JavaScript. Краткие теоретические положения

Объект Form служит контейнером для пересылки данных от пользователя на сервер. Объект Document является контейнером всех форм на странице. Чтобы обратиться к отдельному элементу в конкретной форме, необходимо воспользоваться конструкцией document.Имя Формы.ИмяЭлемента.

Document. Myform.myelement.value = “Мое текстовое поле”;

Рассмотрим основные свойства, методы и события формы

| Характеристика | Название |
|---|---|
| Свойства | action, elements[], length, method, name, target |
| Методы | reset(), submit() |
| События | onReset, onSubmit |

Таблица 1. Свойства, методы и события формы

Свойство **action** задает URL-адрес сценария на сервере, которому отправляется форма:

<form name=”myform”
action=http://www.myserver.com/cgi-bin/registration.asp
metod=”post”>
<!—Объекты формы- - >
</form>

Свойство **elements[]** представляет собой массив, в который включены все элементы формы. Ссылаться на элементы можно по их индексу, который начинается с нуля.

formName.element[2].value

Свойство **length** определяет количество элементов на форме.

Свойство **method** определяет способ отправки формы серверу. На практике чаще всего используются методы GET и POST. В методе GET данные передаются на адрес, указанный в свойстве action, через строку поиска (часть адреса, идущая после знака ?) в URL. В методе POST данные передаются в теле HTML – запроса.

Этот метод следует использовать при передаче серверу большого объема данных.

Свойство **name** ссылается на внутренние имена форм.

Свойство **target** указывает окно назначения браузера, в котором должен отображаться результат отправки формы.

Метод **reset** сбрасывает значения всех элементов формы в исходное сотояние.

Метод **submit** () используется для передачи данных формы.

Событие **onReset** восстанавливает значения по умолчанию в полях формы и возникает при нажатии кнопки Reset или при выполнении метода reset ().

Событие **onSubmit** запускается в момент отправки пользователем данных формы на сервер.

Рассмотрим элементы управления в формах.

**Поля ввода данных** предназначены для ввода однострочной информации. Пример данного элемента:

<input type=”text” name =”CustName” size=40 maxlength=60>

**Поля для паролей** предназначены для ввода паролей. Обычно браузеры вместо вводимых символов на экран выводят звездочки (*). Пример этого элемента:

<input type=”password” name = “pass” size=20>

**Многострочные поля ввода** предназначены для ввода информации, которая не помещается в обычных полях ввода. Чтобы создать текстовое поле из десяти строк по 80 символов в каждом, наберите следующий код:

<textarea name=”Comments” rows = 10 cols=80></textarea>

**Флажки** предназначены для обозначения логических данных, принимающих значение true или false. Пример этого элемента:

<input type =”checkbox” name=”hobby” value=”Литература”>

Использование флажка на форме:

Литература

**Переключатели** предназначены, чтобы пользователь выбрал только один вариант из нескольких возможных. Пример этого элемента:

< input type =”radio” name =”trade”

value =”student”> Студент <br>

Использование переключателя на форме:


Студент

**Списки** также предназначены для выбора элемента. Однако в отличии от переключателей из списка можно выбрать несколько элементов. Пример списка:

<select name =”writer” size=2 multiple>
<option> Иванов </option>
<option> Петров </option>
<option> Сидоров </option>
</select>

Если параметр size=1, то список раскрывающийся. При size=2 список будет прокручиваться внутри окна. Параметр multiple служит для выбора нескольких элементов из списка.

**Кнопки**

В стандарте HTML определены четыре типа кнопок, которые можно включать в формы: Button (обычная кнопка), Submit (кнопка отправки данных на сервер), Reset (кнопка сброса полей формы), Image (кнопка с изображением).

**Задание №20**

**Цель работы:** изучить методику создания формы регистрации на сайте

**Назначение программы:** программа создает типовую форму регистрации на сайте следующего вида

Регистрация на сайте

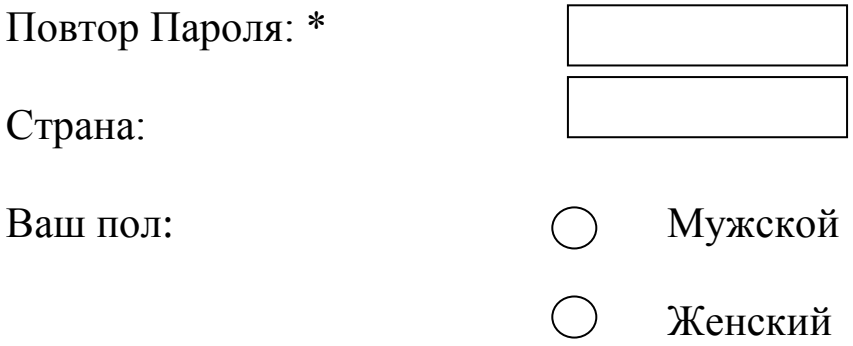
Имя: *

Фамилия: *

Адрес Email: *

Логин: *

Пароль: *




Поля, помеченные звездочкой (*), являются обязательными для заполнения.

Рис. 9 Форма регистрации на сайте.

С помощью функции validateForm ( ) при отправке формы на сервер (см. текст программы) происходит проверка заполнения всех полей формы.

**Текст программы:**

```
<html>
<head>
<title>Регистрация на сайте</title>
<script>
//Функция проверки формы на наичие пустых полей
function validateForm()
{
var validateFlag=true;//возвращаемая функция логическая перемен-ная
for (var i=0; i<6;i++)//Проверяем первые шесть элементов
{
// В действительности переменная countElem не нужна. Она введена для
// улучшения читабельности кода
var countElem=document.regform.elements[i];
if (countElem.value=="") //Текущий элемент пустой?
{//Если да, то
validateFlag=false; //Устанавливаем флаг в false
```

```
//Выдаем предупреждение для текущего элемента
alert ("Введите значение в поле"+countElem.name);
break; // Цикл уже можно не продолжать
}
}
alert('Поздравляем, Вы успешно прошли регистрацию');
return validateFlag;
}
</script>
</head>
<body>
<h2>Регистрация на сайте</h2>
<!--Создаем форму-->
<form name="regform" metod="post" onsubmit="return validate-
Form();">
<!--Для удобства форматирования элементы формы
располагаем в таблице-->
<table width="100%">
<tr>
<!--Создаем поля ввода данных-->
<td> Имя: <sup>*</sup></td>
<!--Поле Имя-->
<td><input type="text"size="35" name="firstname"></td>
</tr>
<tr>
<td>Фамилия: <sup>*</sup></td>
<!--Поле Фамилия-->
<td><input type="text"size="35" name="lastname"></td>
</tr>
<tr>
<td>Адрес Email: <sup>*</sup></td>
<!--Поле Email-->
<td><input type="text"size="35" name="email"></td>
</tr>
<tr>
<td>Логин: <sup>*</sup></td>
<!--Поле Логин-->
```

```
<td><input type="text"size="35" name="login"></td>
</tr>
<tr>
<td>Пароль: <sup>*</sup></td>
<!--Поле Пароль-->
<td><input type="password" size="20" name="pass1"></td>
</tr>
<tr>
<td>Повтор Пароля: <sup>*</sup></td>
<!--Поле Повтор Пароля-->
<td><input type="password"size="20" name="pass2"></td>
</tr>
<tr>
<td>Страна:</td>
<!--Далее список-->
<td><select size="1" name="country">
<!--Вариант списка-->
<option value="Russia">Россия</option>
<option value="Ukraine">Украина</option>
<option value="Belorussia">Белоруссия</option>
<option value="USA">США</option>
<option value="France">Франция</option>
<option value="Great Britain">Великобритания</option>
<option value="other">Другая</option>
</td>
</tr>
<tr>
<td>Ваш пол:</td>
<td>
<!--Формируем переключатели -->
<input type="radio" name="sex" value="male">Мужской<br/>
<input type="radio" name="sex" value="female">Женский<br/>
</td>
</tr>
<tr>
<td colspan=2>
```

```
<!--Флажок со своим описанием занимает всю строку в таблице
-->
<input type="checkbox" name="consent">
С условием регистрации согласен.</td>
</tr>
<tr>
<td colspan=2> Поля, помеченные звездочкой (<sup>*</sup>), яв-ляются обязательными для заполнения.</td>
</tr>
<tr>
<!--Заканчивают форму две стандартные кнопки-->
<td><input type="submit" value="Зарегистрировать"></td>
<td><input type="reset" value="Очистить"></td>
</tr>
</table>

</form>
</body>
</html>
```

**Дополнительные задания:**

1. Усовершенствуйте форму регистрации на сайте таким образом, чтобы происходила проверка на совпадение полей: «пароль» и «повтор пароля».

2. Усовершенствуйте форму регистрации на сайте таким образом, чтобы происходила проверка поля адрес E-mail на наличие символа @.

**Задание №21**

**Цель работы:** изучить методику защиты паролем конкретной страницы сайта.

**Назначение программы:** если пароль неверный, то пользователь переадресовывается, если верный, то выдается сообщение и загружается текущая страница. Данный Java-скрипт следует вставить между тегами <head> и </head> той Web-страницы, которую следует защитить паролем. В данном демонстрационном примере пароль password, в случае неверного

ввода пользователь переадресовывается на Web-страницу http://studia.scriptic.ru/

**Текст программы:**

```
<head>
<title>Защита паролем. Если пароль неверный то пользователь пе-реадресовывается, если верный то выдается сообщение и загружается страница.</title>
<script language=JavaScript>
<!--//
p = prompt('Пожалуйста, введите пароль');
// Запрос пароля
if (p=='password') {
// Не забудьте сменить пароль
alert('Вход разрешен')
// текст сообщения о том что пароль принят
} else {
top.location.href="http://studia.scriptic.ru/"
// Страница на которую будет направлен пользователь если пароль не верен
}
//-->
</script>
</head>
```

**Задание №22**

**Цель работы:** изучить методику защиты паролем сайта, при которой на каждой странице сайта используется отдельный логин и пароль.

**Назначение программы:** В данном демонстрационном примере рассматривается защита сайта из трех страниц. Например, для страницы news.htm используется логин «пример1» и пароль «при-мер1». Данный Java-скрипт может быть реализован на главной странице сайта.

**Текст программы:**

```
< head>
< script language="JavaScript">
<!-- Begin
```

```
function Login(){
var done=0;
var username=document.login.username.value;
username=username.toLowerCase();
var password=document.login.password.value;
password=password.toLowerCase();
if (username=="пример" && password=="пример") { window.location="contakt.htm"; done=1; }
if (username=="пример1" && password=="пример1") { window.location="news.htm"; done=1; }
if (username=="пример2" && password=="пример2"){ window.location="serv.htm"; done=1;}
// и так далее...
if (done==0) { alert("Невеный пароль или имя пользователя!"); }
}
// End -->
</ script>
</head>
<body>
<center>
<form name=login>
<table width=225 border=1 cellpadding=3 bgcolor="#808080">
<tr><td colspan=2>
<p align="center"><b><font color="#FFFFFF"
 size="3"></font></b></p>
</td></tr>
<tr><td>
<font size="3" color="#FFFFFF">
Имя:
</font>
</td><td>
<font size="3" color="#FFFFFF">
<input type=text name=username>
</font>
</td></tr>
<tr><td>
<font size="3" color="#FFFFFF">
```

```
Пароль:
</font>
</td><td>
<font size="3" color="#FFFFFF">
<input type=text name=password>
</font>
</td></tr>
<tr><td colspan=2 align=center>
<font size="3" color="#FFFFFF">
<input type=button value="Перейти на сайт" onClick="Login()">

</font>
</td></tr>
</table>
</form>
</center>
</body>
```

## 10. Использование форм в языке программирования JavaScript

В данном параграфе рассмотрим примеры использования форм в задачах выбора вариантов и передачи информации на сервер.

**Задание №23**

**Цель работы:** изучить методику решения задач выбора вариантов с использованием форм и флажков "checkbox".

**Назначение программы:** программа создает форму, в которой пользователю предлагается выбрать хобби. Программа анализирует ответ пользователя и выдает сообщение, каковы хобби пользователя.

Ваше хобби

☐ - литература

☐ - спорт

☐ - туризм

Рис. 10 Форма выбора варианта с помощью флажков "checkbox"

**Текст программы:**

```
<script>
function raport() {
var num=document.myform.length;
var hob="Ваше хобби";
for (var i=0; i<num-1; i++) {
if (document.myform.hobby[i].checked==true) {
hob=hob+" " + document.myform.hobby[i].value;}
}
window.alert(hob);
}
</script>
<form name="myform">
<input type="checkbox" name="hobby" value="литература">
литература</br>
<input type="checkbox" name="hobby" value="спорт">
спорт</br>
<input type="checkbox" name="hobby" value="туризм">
туризм</br>
<input type="button" value="Отправить" onClick="raport()">
</form>
```

**Задание №24**

**Цель работы:** изучить методику передачи информации на сервер с помощью формы.

**Назначение программы:** программа передает на сервер следующую информацию: e-mail пользователя и сообщение. Но поскольку программы ―anyprogramm.php”, которая обрабатывает информацию на сервере не существует, при нажатии кнопки **«Отправить»** будет выдано сообщение ―Невозможно найти удалѐнный сервер”.

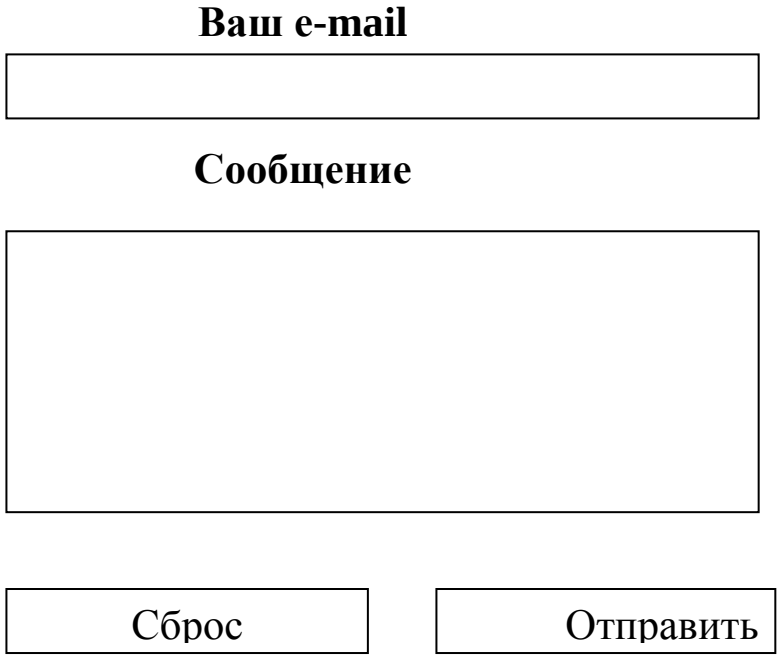



Рис. 11 Форма отправки сообщения на сервер

**Текст программы:**

```
<html>
<head>
<title>Форма</title>
<style type="text/css">
form
{
width:300px;
padding:20px;
border-with:lpx;
border-style:solid;
background:#e0e0e0
}
</style>
</head>
<body>
<form action="http://anyserver.ru/anyprogramm.php" method="post">
<h2>Сообщение автору </h2>
Ваш E-mail:
```

```
<input type="text" name="from" value="" size=30/>
<br/>
Сообщение: <br>

<textarea name='text'cols=30 rows=8></textarea>
<br/>
<input type="reset"/>
<input type="submit" name="Отправить" value="Отправить"/>
</form>
</body>
</html>
```

**Задание №25**

**Цель работы:** получить навыки самостоятельной разработки программ для решения задач выбора варианта с использованием форм и переключателей “radio”.

**Назначение программы:** программа реализует тестовый вопрос, в котором из 4 возможных вариантов ответов надо выбрать один правильный. Программа должна проанализировать ответ пользователя и выдать сообщение:

1. Ответ правильный.

2. Ответ неправильный. Вам надо прочитать теоретический материал по сетевым атакам.

Какая сетевая атака связана с превышением допустимых пределов функционирования сети?

◯ Подслушивание (Sniffing).

◯Отказ в обслуживании (DoS –атака).

◯ Атака Man in – the – Middle (человек в середине).

◯ Угадывание ключа.

Рис. 12 Форма выбора варианта с помощью переключателей “radio”.

## 10. Слои в языке программирования JavaScript. Краткие теоретические положения

Для создания слоев используется элемент DIV, являющийся блочным элементом разметки. Рассмотрим свойства, указывающие способ размещения и точное позиционирование слоев. Свойство position указывает способ размещения элементов. Оно может принимать четыре значения:

absolute – задает абсолютное свободное позиционирование;

fixed – аналогично absolute, но при прокрутке элемент не перемещается;

relative – определяет относительное свободное позиционирование;

static – задает статическое позиционирование, при котором элемент страницы отображается внутри общего «потока» текста.

Свойства left, right, top и bottom определяют точное позиционирование элементов:

left – задает горизонтальную позицию левой границы свободно позиционированного элемента;

right – определяет горизонтальную позицию правой границы свободно позиционированного элемента;

top – задает вертикальную позицию верхней границы свободно позиционированного элемента;

bottom – определяет вертикальную позицию нижней границы свободно позиционированного элемента.

Например, следующий фрагмент программы выводит слой в правом нижнем углу:

<div id=third style=”position:absolute; bottom:50px;right:50px;

Background-color:red;”>

Слой в правом нижнем углу

Свойства height и width определяют высоту и ширину блока, определяющего слой.

Свойство visibility используется для того, чтобы делать слои видимыми и скрытыми. Оно может принимать следующие значения:

hidden – скрывает элемент;

collapse – если не используется с таблицами, то аналогично значению hidden;

inherit – значение по умолчанию, которое указывает, что видимость наследуется от родителя;

visible – делает элемент видимым.

**Задание №26**

**Цель работы:** изучить методику использования и позиционирования слоев.

**Назначение программы:** программа выводит на экран три слоя, причем третий слой является прозрачным, поэтому сквозь него виден второй слой.

**Текст программы:**

```
<html>
<head>
<title>Перекрывающиеся слои</title>
</head>
<body>
<h2>Перекрывающиеся слои </h2>
<div id=first style="position:absolute; top:50px;left:50px;
height:150px; width:150px; background-color:orange;">
1 слой
</div>
<div id=second style="position:absolute; top:70px; left:70px;
height:150px; width:150px; background-color:red;">
2 слой
</div>
<div id=third style="position:absolute; top:90px;left:90px;
height:150px; width:150px;">
3 слой
</div>
</body>
<html>
```

**Задание №27**

**Цель работы:** изучить методику управления видимостью слоев.

**Назначение программы:** программа создает невидимый слой, который можно отобразить или вновь скрыть щелчком кнопки мыши по ссылке.

**Текст программы:**

```
<html>
<head>
<title> Управление видимостью </title>
<script>
//Функция, осуществляющая изменение видимости слоя
function vis(){
//Получаем доступ к слою
var Layer=document.getElementById('lay');
//выясняем состояние слоя
if (Layer.style.visibility=='hidden') {
//если слой невидим, то делаем видимым
Layer.style.visibility='visible';
}
else{
//если слой видим, то делаем невидимым
Layer.style.visibility='hidden';
}
}
</script>
</head>
<body>
<h2> Упраление видимостью </h2>
<div id=control style="position:absolute; top:50px;left:50px;">
<a href="javascript:vis();">Скрыть/Показать слой </a>
</div>
<div id=lay style="position:absolute; top:70px;left:50px;
height:100px;width:250px; background-color:orange;
visibility:hidden">

Этот слой по умолчанию невидим. Чтобы его скрыть или увидеть,
необходимо щелкнуть кнопкой мыши на ссылке
</div>
```

```
</body>
</html>
```

## 12. Защита контента от несанкционированного копирования информации в языке программирования JavaScript. Краткие теоретические положения

Рассмотрим 3 варианта Java-скриптов, предназначенных для защиты контента сайта от несанкционированного копирования. Java-скрипты необходимо вставлять между тегами <head> и </head>, например после тега </title>.

1. Рассмотрим Java-скрипт, позволяющий скопировать текст, но после вставки текста в конце появляется ссылка на источник, например: Источник: © www.swsu.ru

```
<script type="text/javascript">
function addLink() {
var body_element = document.getElementsByTagName('body')[0];
var selection;
selection = window.getSelection();
var pagelink = "<p>Источник:
 <a href="'+document.location.href+'">"+"</a> © www.swsu.ru";
var copytext = selection + pagelink;
var newdiv = document.createElement('div');
newdiv.style.position='absolute';
newdiv.style.left='-99999px';
body_element.appendChild(newdiv);
newdiv.innerHTML = copytext;
selection.selectAllChildren(newdiv);
window.setTimeout(function() {
body_element.removeChild(newdiv);
},0);
}
document.oncopy = addLink;
</script>
```

2. Рассмотрим второй Java-скрипт, который не только запрещает выделять текст на странице, но и пользоваться правой кнопкой мышки.

```
<script language="JavaScript">
```

```
document.onselectstart=function(){return false}
document.oncontextmenu=function(){return false}
document.onmousedown=function(){return false}
</script>
```

3. Рассмотрим третий Java-скрипт, который защищает контент от несанкционированного копирования прозрачным слоем. При попытке копирования пользователь копирует прозрачный слой.

```
<html>
<head>
<title>Издательство НТ Пресс</title>
<div style="position: relative;">
//далее защищаемый прозрачной картинкой контент сайта
<img style="position: absolute; top: 0pt; left: 0pt; width: 100%; height: 100%;"
src="transparent_gif.gif" alt="" />
</div>
</head>
<body>
</body>
</html>
```

**Задание №28**

**Цель работы:** изучить методику защиты контента сайта от несанкционированного копирования

**Назначение программы:** программа реализует сайт, состоящий из трех страниц. Сайт на произвольную тему разработайте самостоятельно или используйте сайт из практического задания №8. Каждую страницу сайта защитите от несанкционированного копирования разными видами Java-скриптов, рассмотренными в кратких теоретических положениях.

## Список контрольных вопросов

1. Какие методы ввода и вывода информации существуют в языке JavaScript? Поясните на примерах.

2. Нужно ли при объявлении переменной в языке JavaScript указывать ее тип?

3. Каким образом можно вывести значение всех элементов массива в языке JavaScript?

4. Какие операторы языка JavaScript служат для реализации механизмов ветвления. Запишите общий вид этих операторов.

5. Какие операторы языка JavaScript служат для реализации механизмов цикла. Запишите общий вид этих операторов.

6. Дайте определение функции и запишите ее общий вид в языке JavaScript.

7. Для какой цели в языке JavaScript используется оператор return?

8. Приведите примеры двух-трех событий и соответствующих им обработчиков этих событий.

9. Каким образом можно программным путем можно обратиться к полю формы и присвоить ему определенное значение? Приведите пример.

10. Каким образом программа определяет адрес URL-сценария на сервере по которому передаются данные формы?

11. Какие методы передачи данных формы на сервер существуют и в чем их различие?

12. Дайте краткую характеристику основных элементов управления, используемых в формах ( поля ввода данных, поля ввода паролей, флажки, переключатели, списки)

13. С какой целью при работе с формами используются метод Reset и событие onReset?

14. Какие существуют методы защиты контента сайта от несанкционированного копирования информации. В чем их различие?

## Список литературы

1. Дунаев В. В. Сценарии для Web-сайта: PHP и JavaScript. – 2-е изд., перераб. и доп. – СПБ.: БХВ – Петербург, 2008. -576 с.

2. Днепров А. Г. JavaScript на 100%. – СПб.: Питер, 2008. -304 с.

3. Колисниченко Д.Н. Современный сайт на PHP и JavaScript. - СПб.: Питер, 2009. – 176 с.

4. Дронов В.А. JavaScript и AJAX в Web-дизайне. - СПб.: БХВ-Петербург, 2008. – 728 с.

5. Прохоренок Н.А., Дронов В. А.: HTML, JavaScript, PHP и MySQL. - СПб.: БХВ-Петербург, 2015. – 766 с.

6. Гарнаев А.Ю. Web-программирование на Java и JavaScript. - СПб.: БХВ-Петербург, 2005. – 1040 с.

7. Дуванов А.А. Web-конструирование. DHTML. - СПб.: БХВ-Петербург, 2003. – 512 с.

8. Кисленко Н. П. HTML. Самое необходимое. БХВ - Петербург, 2008. – 353с.

9. Гаевский А. Ю., Романовский В. А. 100% самоучитель по созданию Web – страниц и Web-сайтов. HTML и JavaScript. - Триумф, 2008. -454 с.