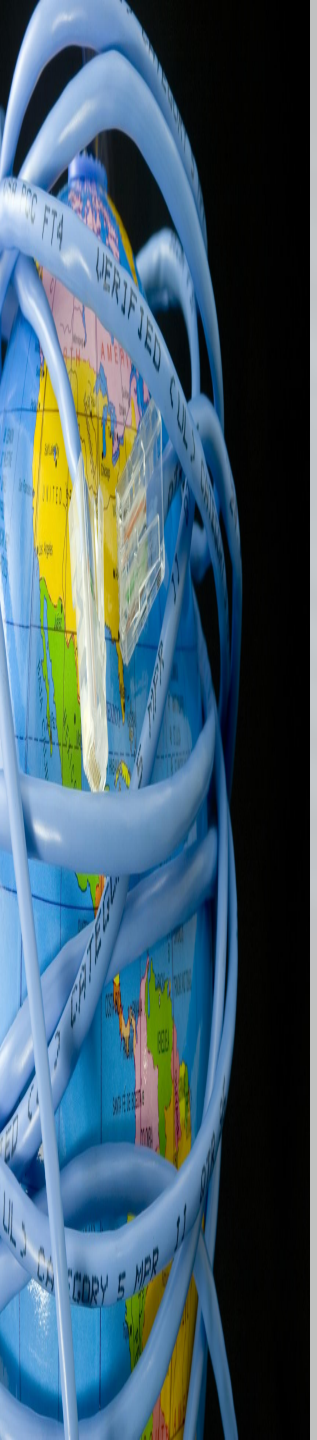
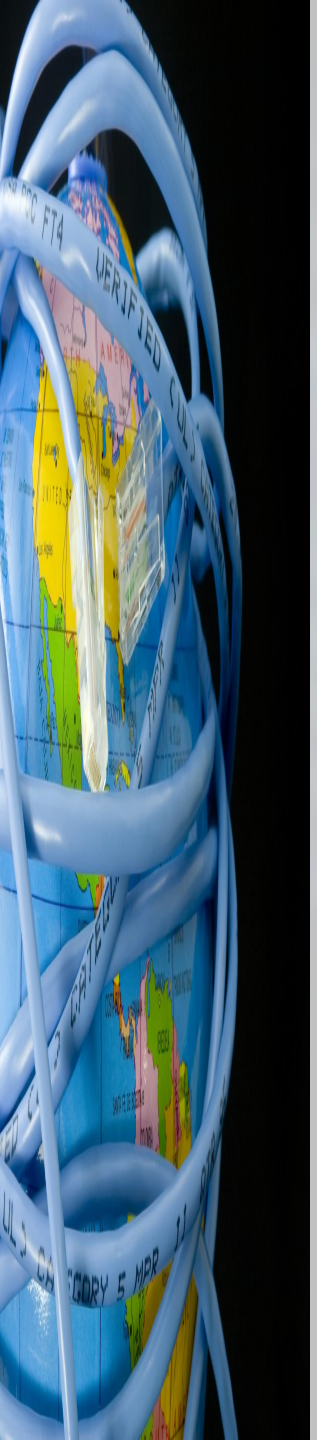


Башкирский государственный университет

Современные Web-технологии

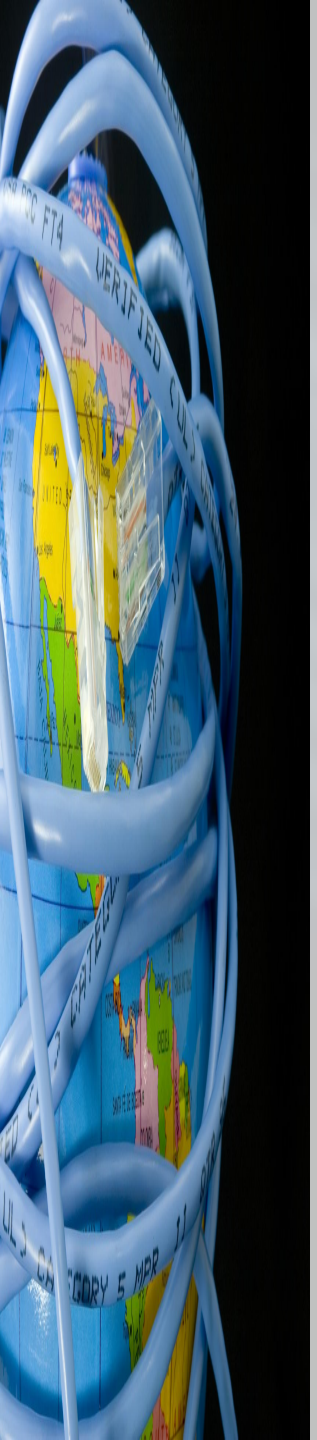
к.т.н., доц. Полупанов Д.В.

Лекция 5. Объектно-событийный язык сценариев *JavaScript*

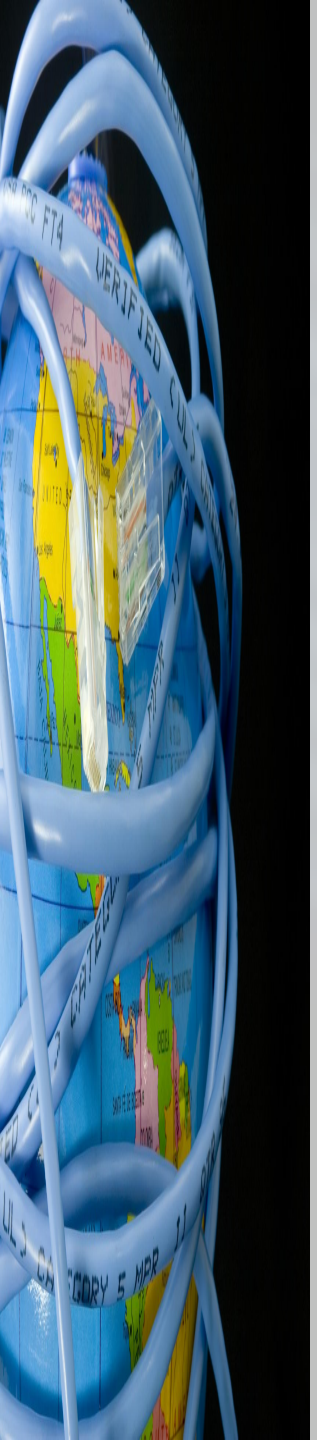


Базовые определения

- **JavaScript** – это объектно-событийный язык сценариев. В этом определении содержатся три ключевых понятия: **сценарий**, **событие** и **объект**.
- **Сценарий** (скрипт, script) – программа, которая расширяет возможности языка HTML. Скрипты обрабатываются (интерпретируется) браузером одновременно с кодом HTML.
- **Событие** – действие пользователя или операционной системы, которое вызывает запуск скрипта (программы). В качестве примера произошедших событий можно назвать: нажатие клавиши на клавиатуре, щелчок мышью, открытие окна браузера, срабатывание таймера.

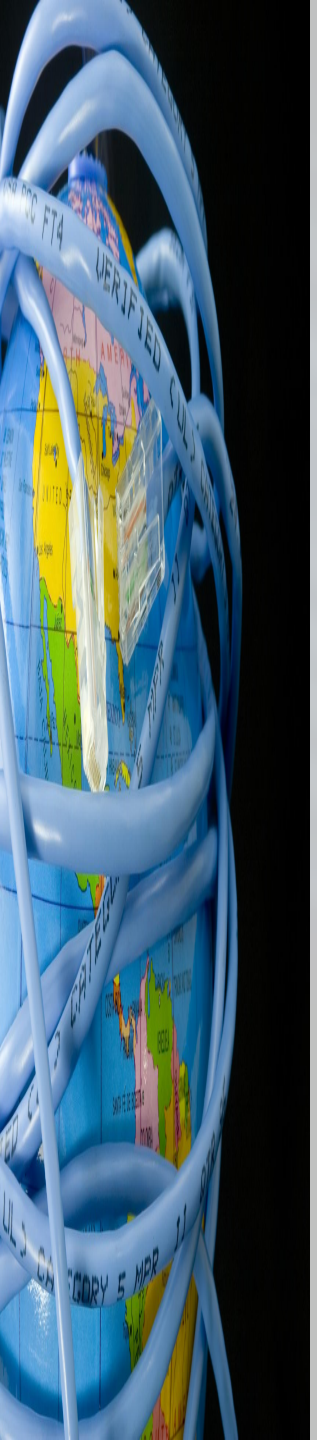


- **Объект** – контейнер, содержащий данные. Объект обладает свойствами и методами, предназначенными для изменения этих свойств.
- Объектами являются:
  - *окно браузера* (*window*);
  - *web-страница* (*document*);
  - *фрейм* (*frame*);
  - *текстовое поле* (*text*);
  - *кнопка* (*button).*
- Часть объектов находятся в состоянии соподчиненности друг к другу (иерархии). Старшие объекты включают в себя младшие объекты. Например, web-страница (document) может содержать внутри себя форму (form), которая может быть реализована в виде текстового поля (text).
- **Свойства** – совокупность переменных, которые характеризуют объект, например, ширина и высота окна браузера.
- **Методы** – функции (подпрограммы), которые позволяют изменять свойства объекта (а значит и сам объект). Характерной синтаксической особенностью методов является то, что их имена заканчиваются скобками.



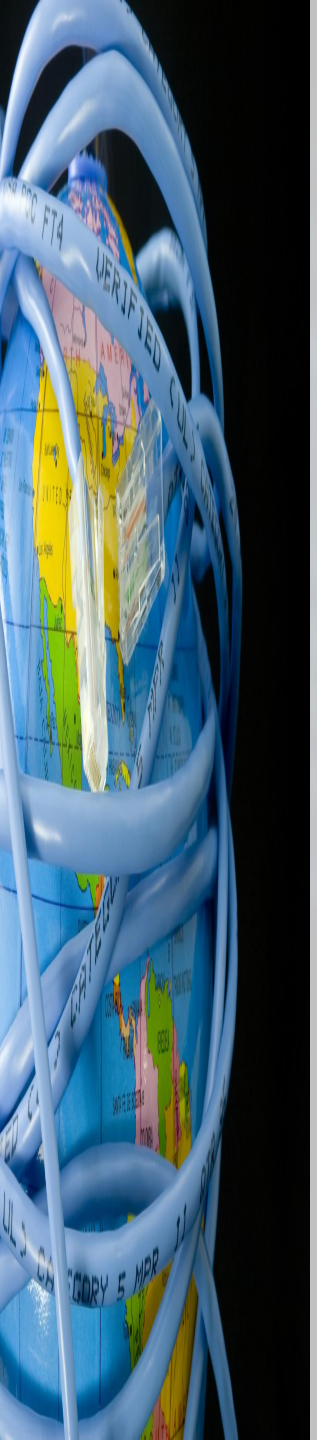
JavaScript не предназначен для создания автономных приложений. Программа на JavaScript встраивается непосредственно в исходный текст HTML-документа и интерпретируется браузером по мере загрузки этого документа.

Первоначальное название - **LiveScript**. JavaScript не имеет никакого отношения к языку **Java**. Java разработан фирмой SUN. JavaScript - фирмой Netscape Communication Corporation. После завоевания языком Java всемирной известности LiveScript из коммерческих соображений переименовали в JavaScript.



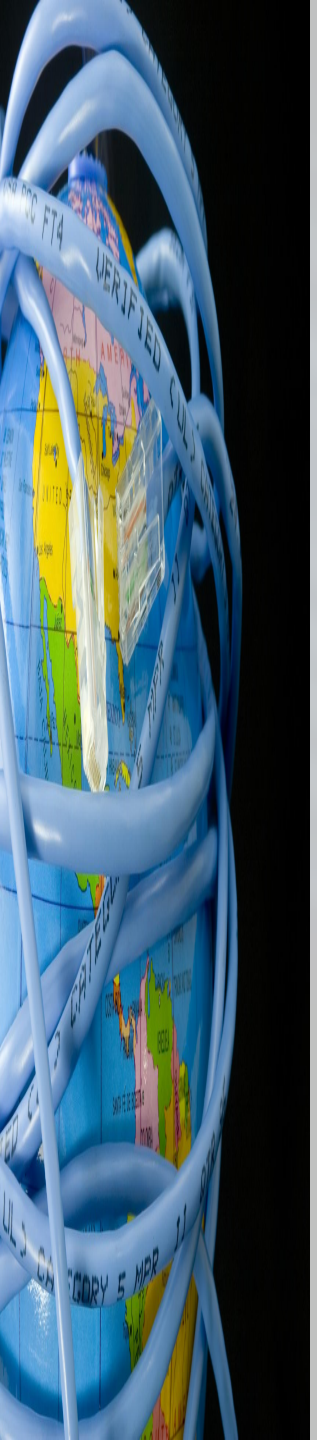
Некоторые особенности

- Язык JavaScript существенно отличается от алгоритмических языков. Работа программ, написанных на алгоритмических языках, идет практически непрерывно от момента их запуска до момента завершения.
- Скрипты – это множество программ, которые работают чаще всего независимо друг от друга и каждая программа запускается автономно при определенных действиях пользователя. По этой причине этот язык иногда называют интерактивным.
- При отсутствии активных действий пользователя скрипты не будут работать (за исключением, может быть, сценариев, которые запускаются с помощью таймера или в момент полной загрузки Web-страницы). Большую часть времени скрипты «дремлят», ожидая активных действий пользователя.



Элементы языка

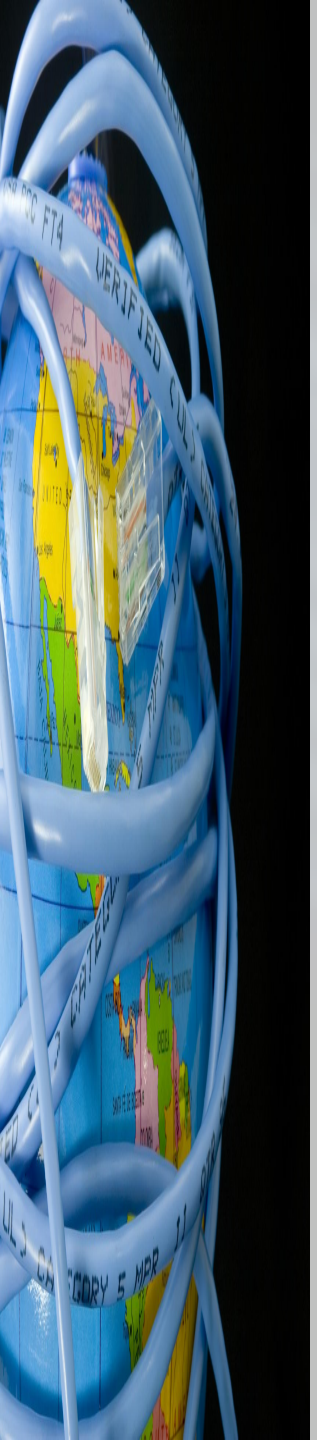
- константы;
- переменные;
- функции;
- операторы;
- выражения.



Константы

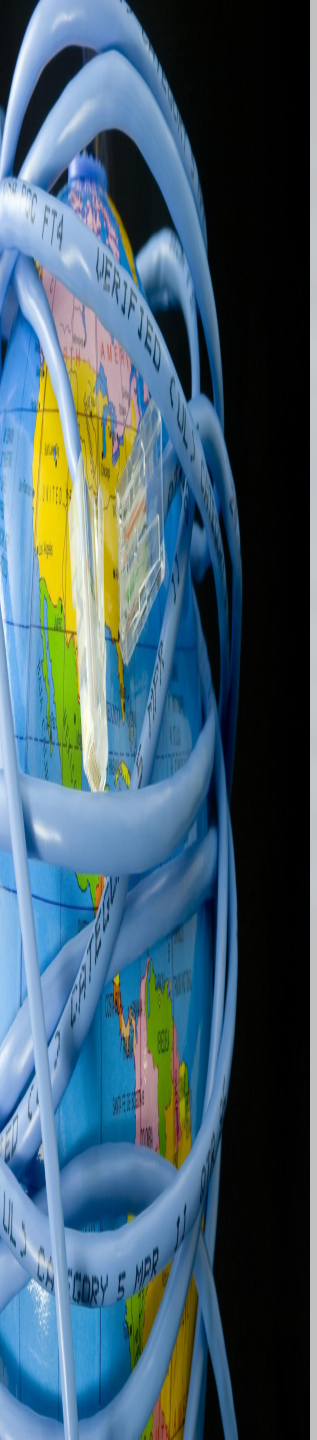
- Это величины, которые не изменяют своих значений в процессе работы скрипта.
- Константы бывают следующих типов:

- целые (десятичные, восьмеричные, шестнадцатеричные);
- с плавающей точкой;
- строковые;
- булевы (логические);
- null, означающая отсутствие каких-либо значений.



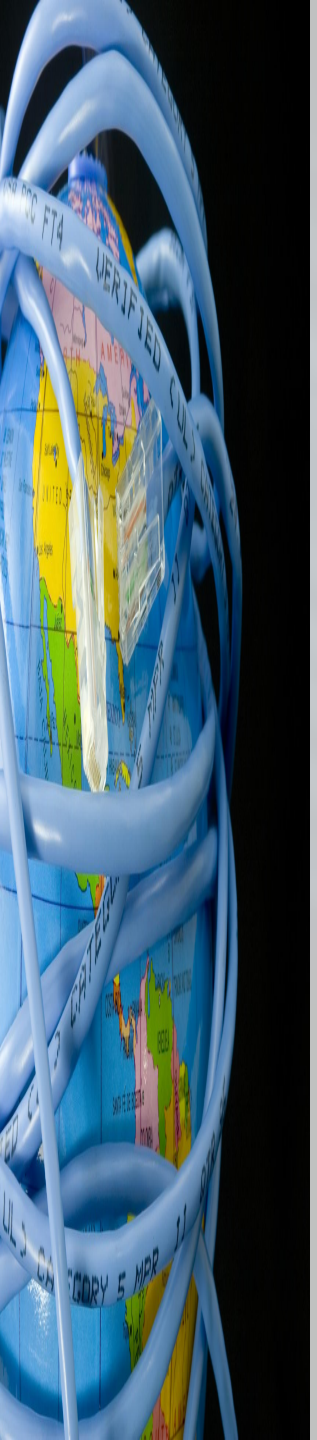
Переменные

- Это величины, которые принимают различные значения в процессе работы скрипта.
- Переменные могут быть *целыми*, *с плавающей точкой*, *строковыми* и *булевыми*. Перед использованием переменные должны быть объявлены.
- Переменные могут быть *глобальными* и *локальными*. Локальные действуют только внутри функции, глобальные действуют в рамках всей составленной программы. Однако и глобальные переменные действуют лишь в пределах одной Web-страницы.
- Имена переменных должны начинаться с буквы или с символа подчеркивания. В именах допустимо использование арабских цифр. Имена не могут содержать пробелов. Они не должны совпадать с ключевыми (зарезервированными) словами языка JavaScript.



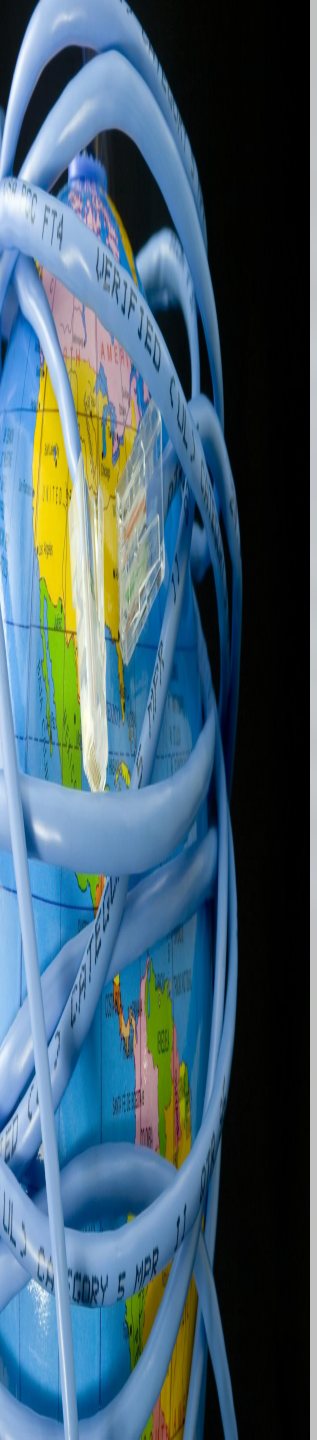
Функции

- **Функция** – логически завершенный фрагмент программы, предназначенный для решения (реализации) определенной задачи.
- Каждая функция должна иметь оригинальное имя.
- Одна и та же функция может быть многократно вызвана в разных местах программы.



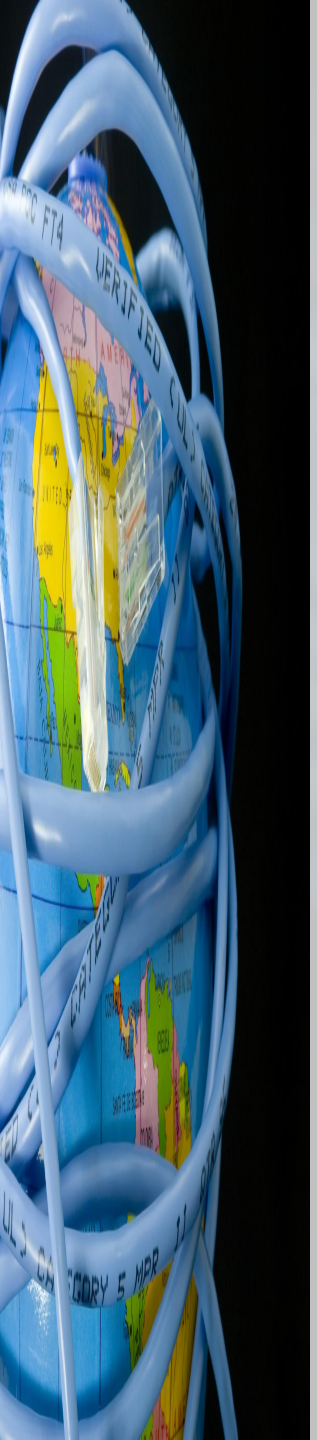
Операторы

- Это конструкции, которые определяют, какие действия производятся над константами, переменными, функциями.
- Операторы бывают следующих типов:
  - присваивания;
  - сравнения;
  - арифметические;
  - инкремента и декремента;
  - битовые;
  - логические;
  - строковые;
  - специальные;
  - управляющие;
  - цикла.



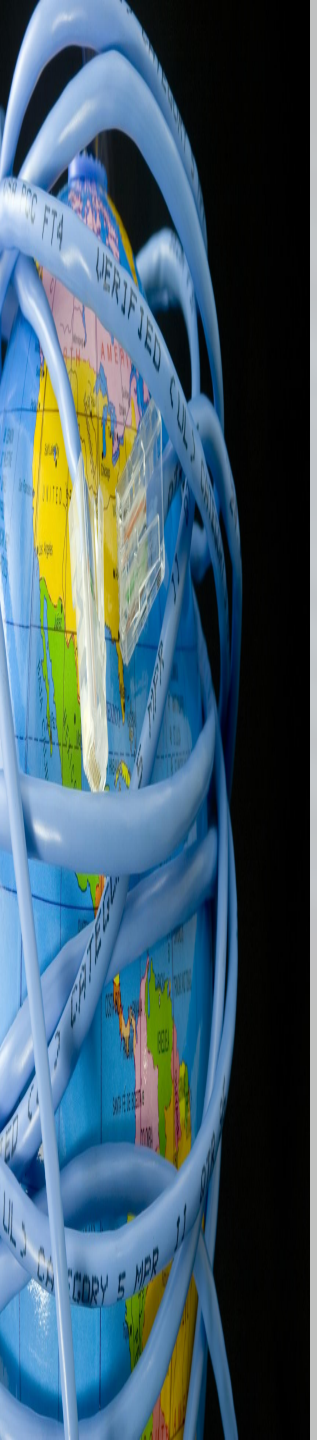
Выражения

- Это набор констант, переменных, функций, соединенных операторами.
- Каждое выражение в языке JavaScript должно заканчиваться точкой с запятой.



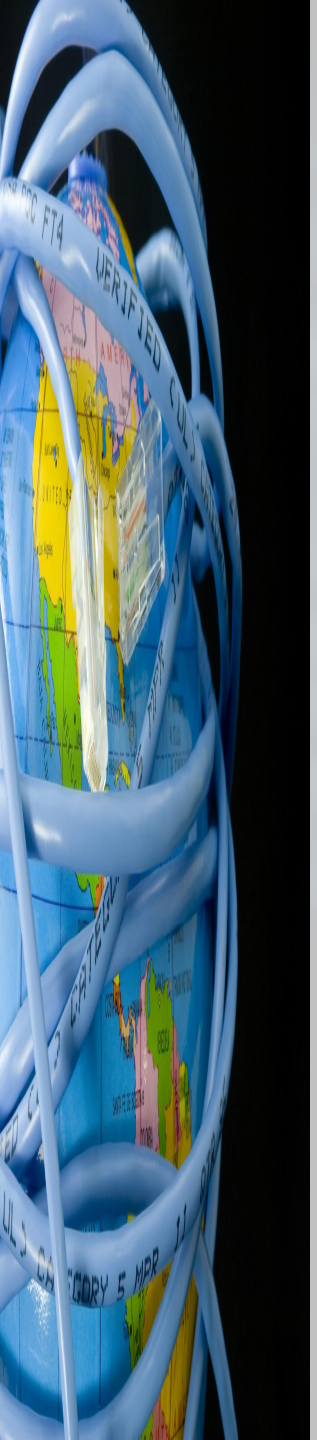
Важно

- Программа, написанная на языке JavaScript, чаще всего размещается внутри HTML-кода, например, внутри контейнера из тегов **head**.
- Границы сценария отмечают при помощи парного тега **script**.
- Существует возможность разместить код скрипта в отдельном файле. В этом случае можно вызывать один и тот же сценарий на разные HTML-страницы.



Начинаем программировать

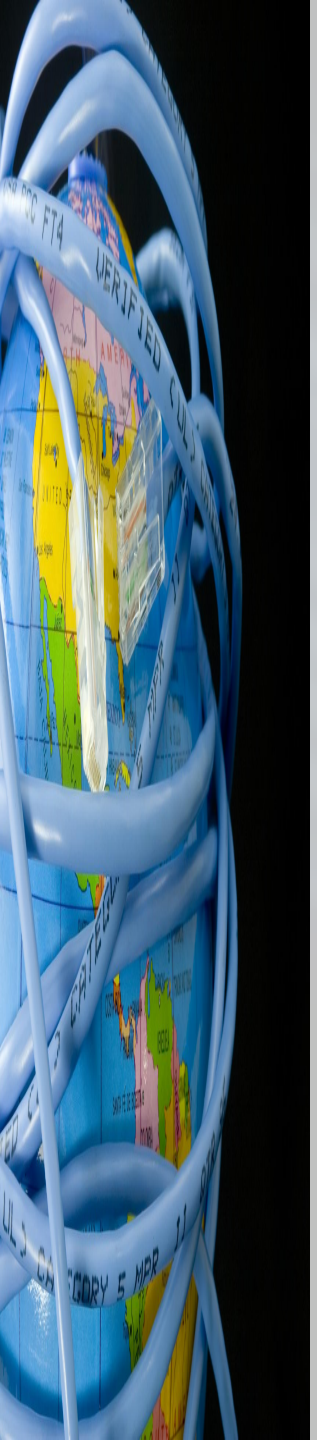
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<head>
 <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
</head>
<body>
 <h1>Начнем?</h1>
 <script type="text/javascript">
 <!–
document.write("Привет!");
 //-->
 </script>
 </body>
</html>
```



Комментарии к программе

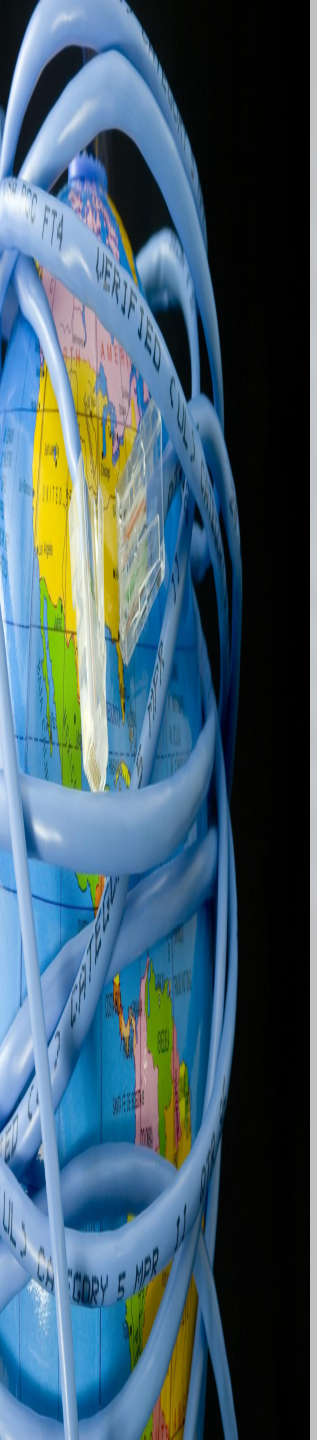
- Текст сценария оформляется как комментарий, чтобы не было проблем у посетителей, браузеры которых не понимают **JavaScript**. Кроме того к символам, завершающим комментарий добавляется еще два символа "**/**", т.к. некоторые браузеры рассматривает строку, состоящую только из символов "**-->**", как ошибочную.
- В примере для объекта с именем **document** вызывается метод **write**. В качестве параметра ему передается текстовая строка "Привет!". Строка закрывается символом "**;**", которым отделяются друг от друга все операторы JavaScript.
- Объект **document** – это HTML-документ, загруженный в окно браузера. Метод **write** записывает в тело HTML-документа строку "Привет!". При этом документ будет выгдядеть так,

Имейте в виду, что JavaScript различает строчные и прописные буквы. Кроме того символ дефиса в JavaScript распознается как минус, т.е. если фон объекта в HTML-документе задается через свойство **background-color**, то в JavaScript - через **backgroundColor**.



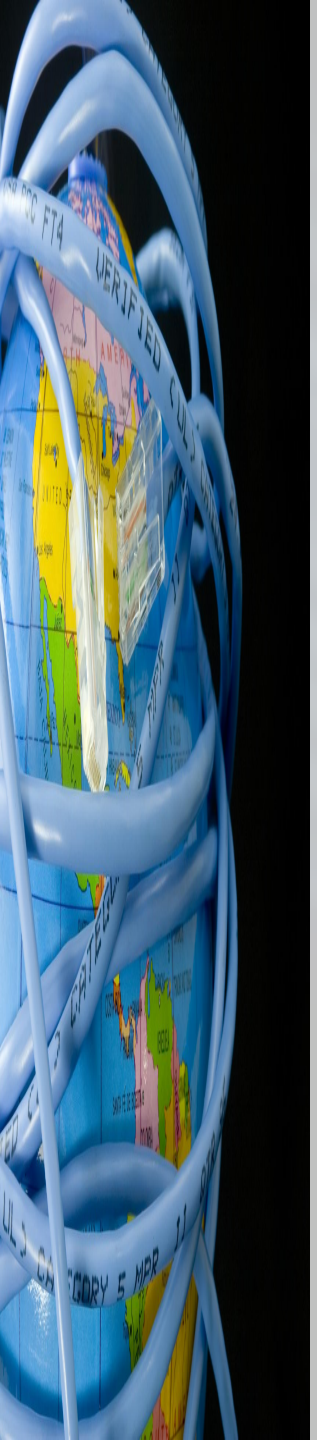
Операторы языка. Унарные операторы

-	Изменение знака на противоположный
!	Дополнение. Используется для реверсирования значения логических переменных
++	Увеличение значения переменной. Может применяться и как префикс, и как суффикс
--	Уменьшение значения переменной. Может применяться и как префикс, и как суффикс



Бинарные операторы

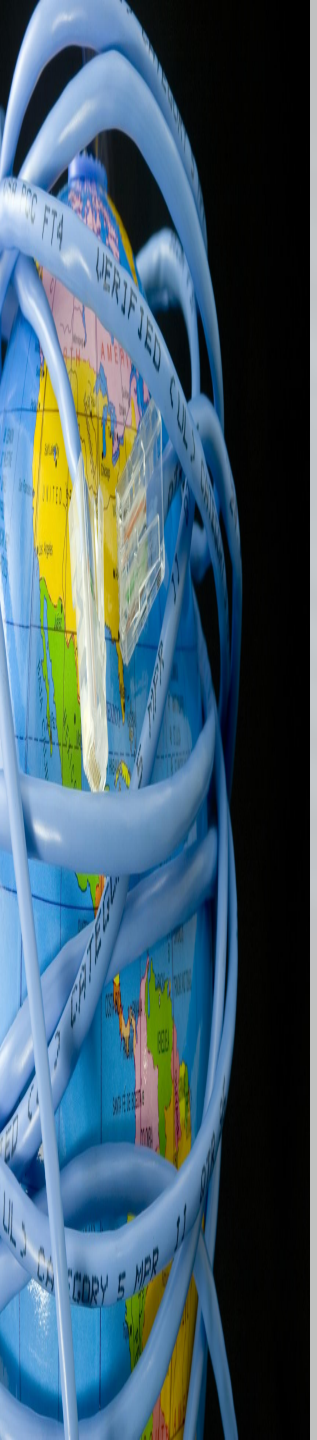
-	Вычитание
+	Сложение
*	Умножение
/	Деление
%	Остаток от деления



Операторы для работы с отдельными битами

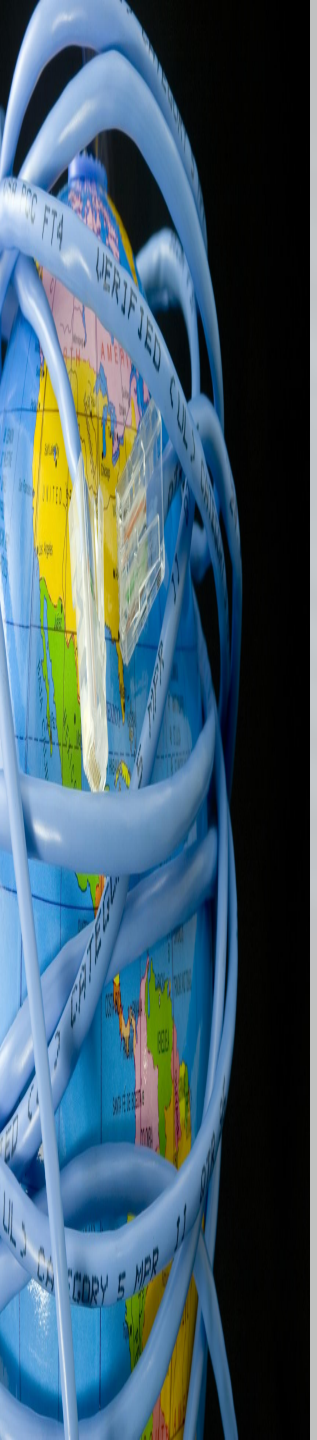
&	И
|	ИЛИ
^	ИСКЛЮЧАЮЩЕЕ ИЛИ
~	НЕ

В условных операторах также применяются логические операторы: || (ИЛИ) и && (И).



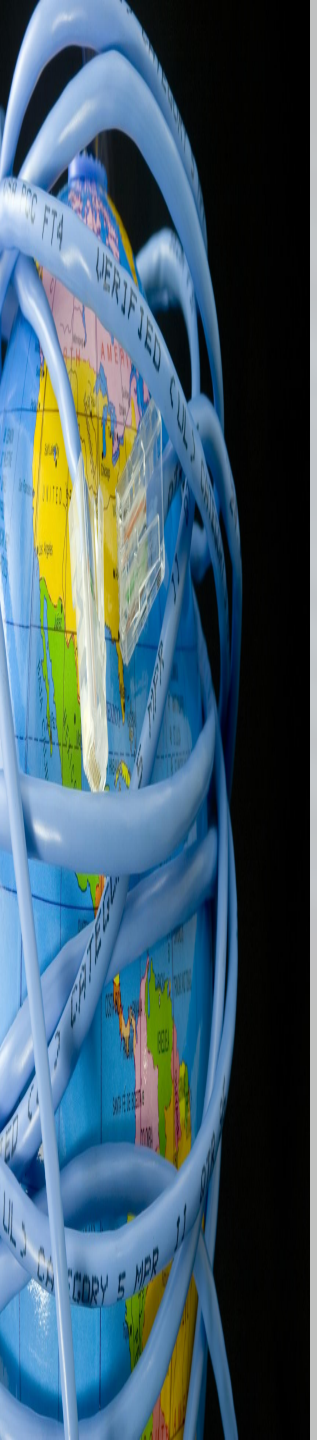
Операторы сдвига

>>	Сдвиг вправо
<<	Сдвиг влево
>>>	Сдвиг вправо с заполнением освобождаемых разрядов нулями



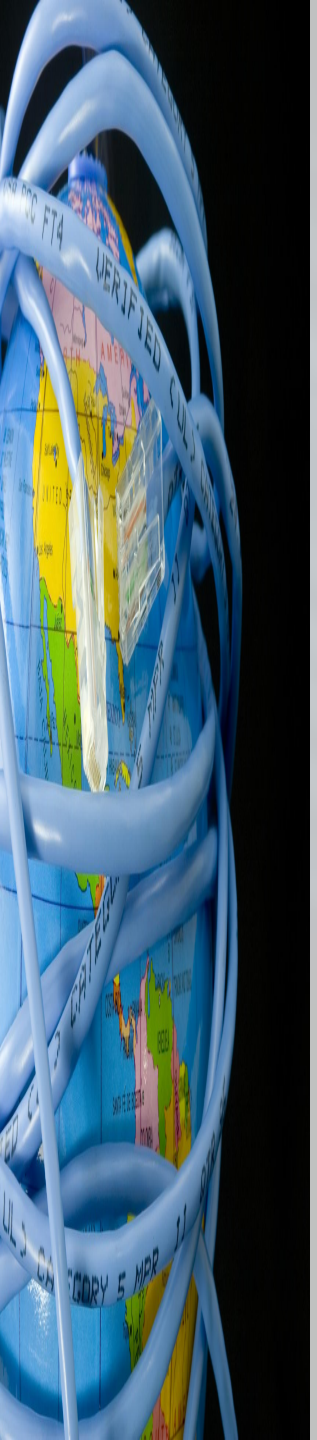
Операторы отношения

>	Больше
>=	Больше или равно
<	Меньше
<=	Меньше или равно
==	Равно
!=	Не равно



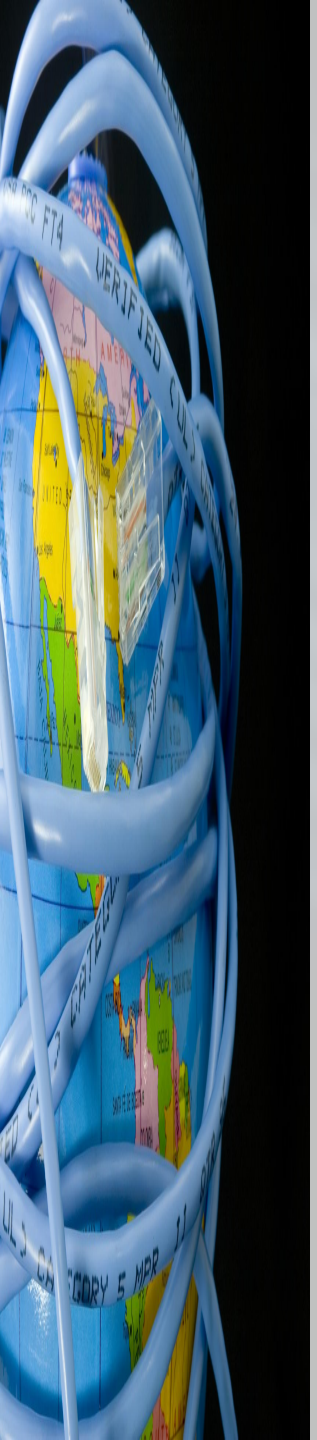
Оператор присваивания и его комбинации

=	Присваивание
+=	Сложение или слияние строк (n=n+7; аналог. n+=7;)
–=	Вычитание (n=n-7; аналог. n-=7;)
*=	Умножение
/=	Деление
>>=	Сдвиг вправо
<<=	Сдвиг влево
>>>=	Сдвиг вправо с заполнением освобождаемых разрядов нулями
&=	И
|=	ИЛИ
^=	ИСКЛЮЧАЮЩЕЕ ИЛИ



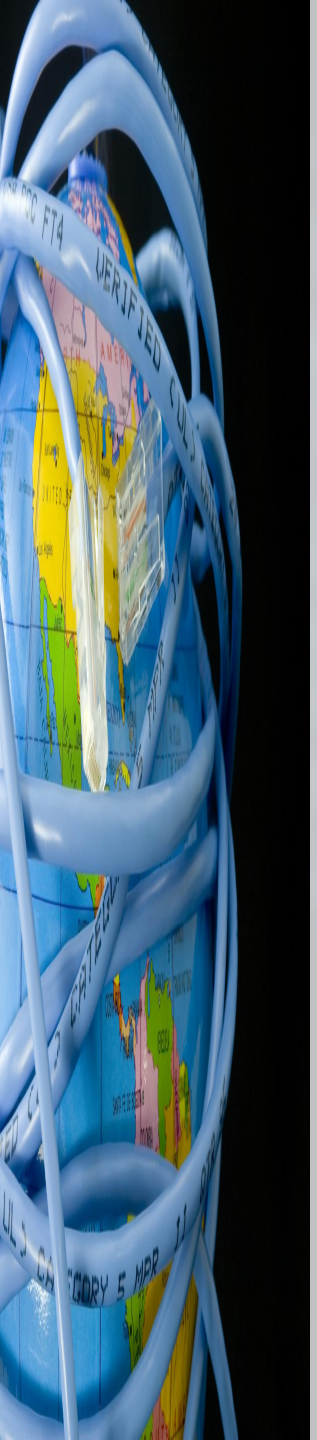
Условные операторы

- if-else
- ?:
- switch - case:



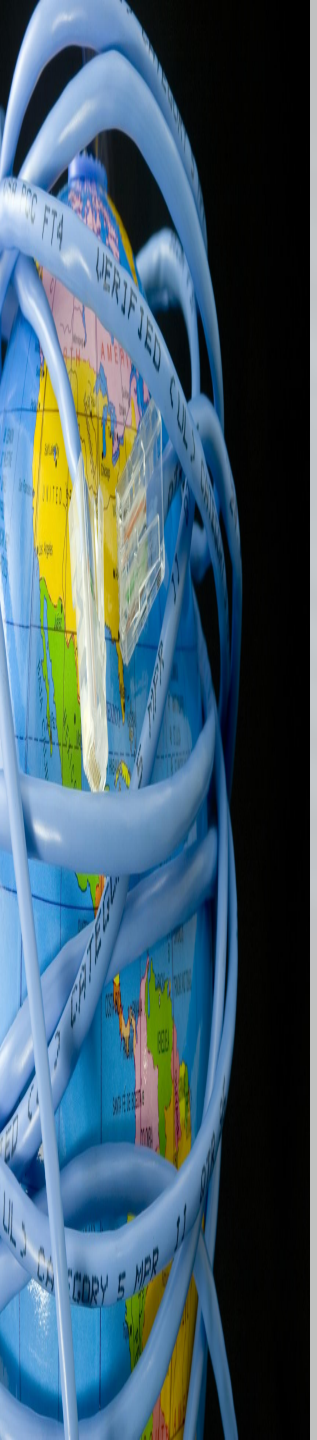
Пример оператора IF-ELSE

```
if(Vol<2)
{
b=true;
ss="w002.htm";
}
else if(Vol>100)
 {
b=true;
ss="w100.htm";
}
```



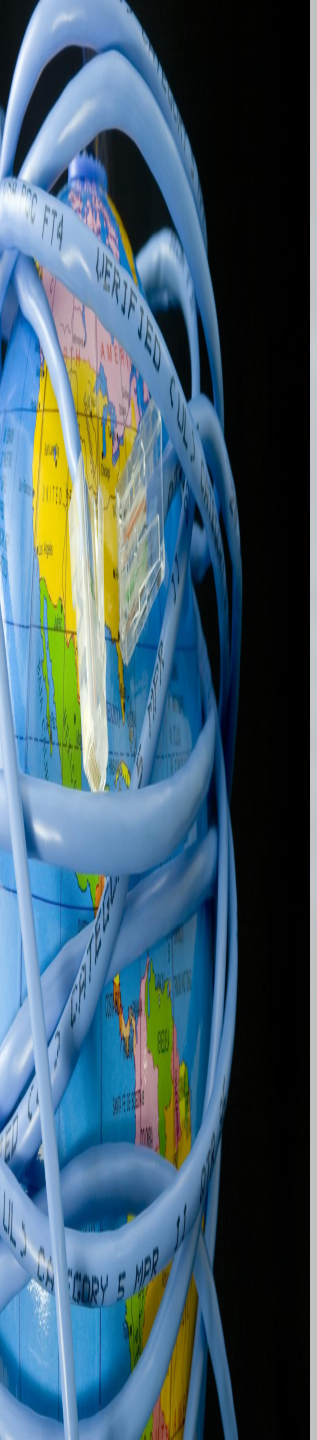
Пример оператора SWITCH - CASE:

```
switch(variable) {
      case value_1: {
           //блок операторов_1
           break;
         }
      case value_2:  {
           //блок операторов_2
       break;
       }
       case value_n: {
           //блок операторов_n
           break;
        }
       default:
            { //блок операторов по умолчанию
       }
}
```



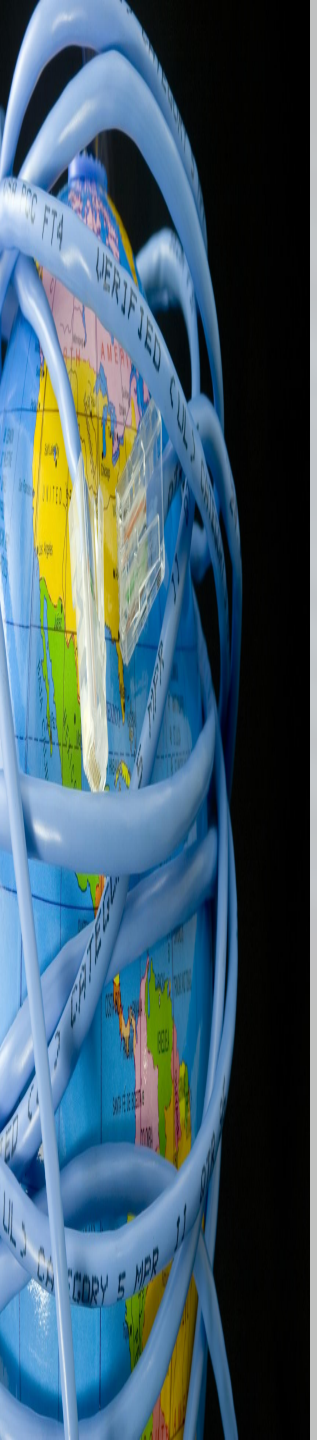
Пример оператора ?:

```
b = (Vol<2 || Vol>100) ? true:false;
```



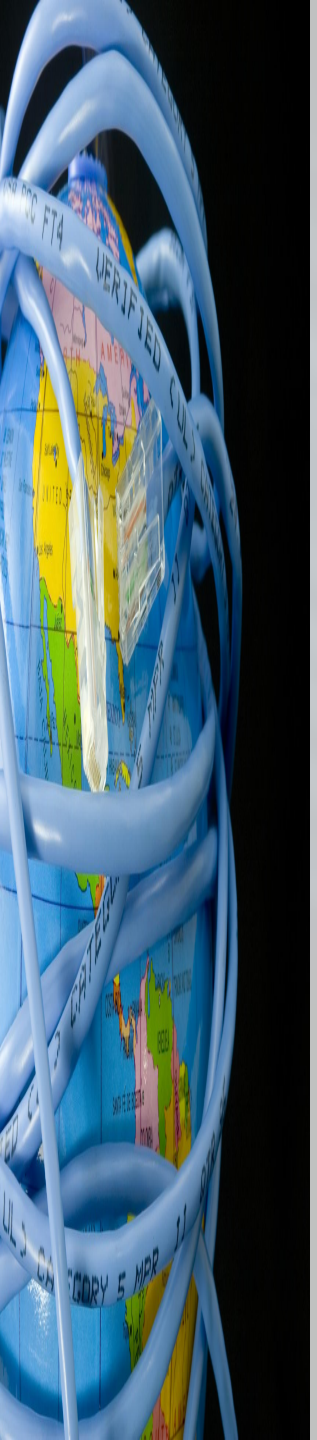
Операторы цикла

- **for**
- **for-in**
- **while**
- Кроме этих операторов в организации цикла могут участвовать еще два оператора: **break** (выход из цикла) и **continue** (переход на следующий шаг).



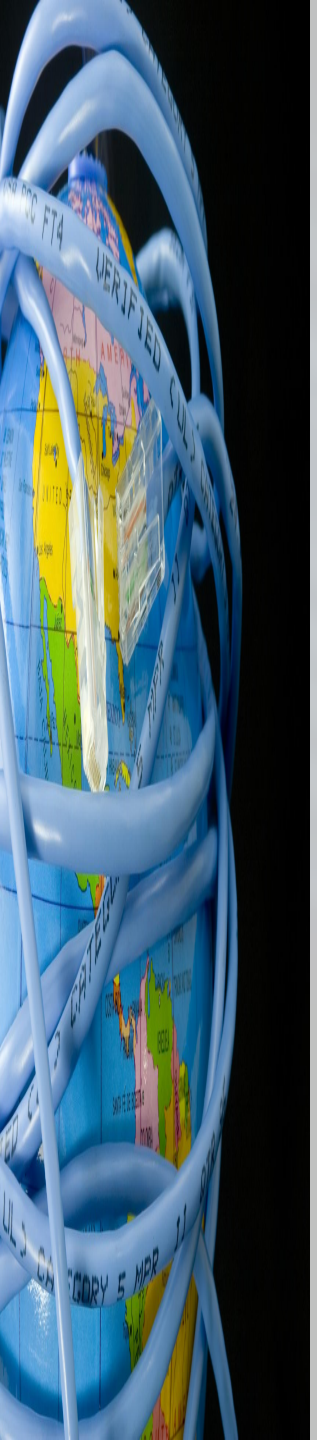
Пример оператора FOR

```
for(i=0; i<n; i++)
 {
     text+=" ";
 }
```



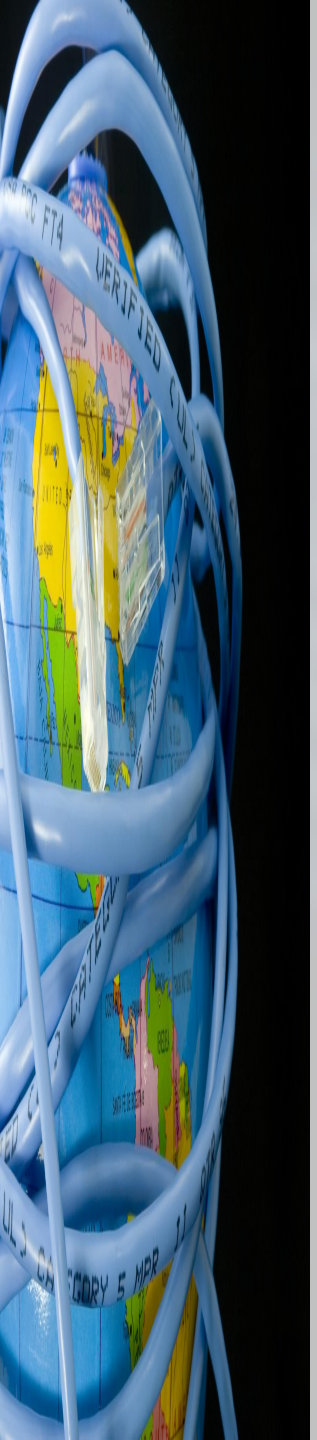
Пример оператора FOR-IN

```
var sprops="<H2>Свойства объекта window</H2>"
        for (props in window)
                sprops+="<b>"+props+"</b><xmp>"+":
"+window[props].substr(0,90)+"</xmp><br>";
document.write(sprops);
```



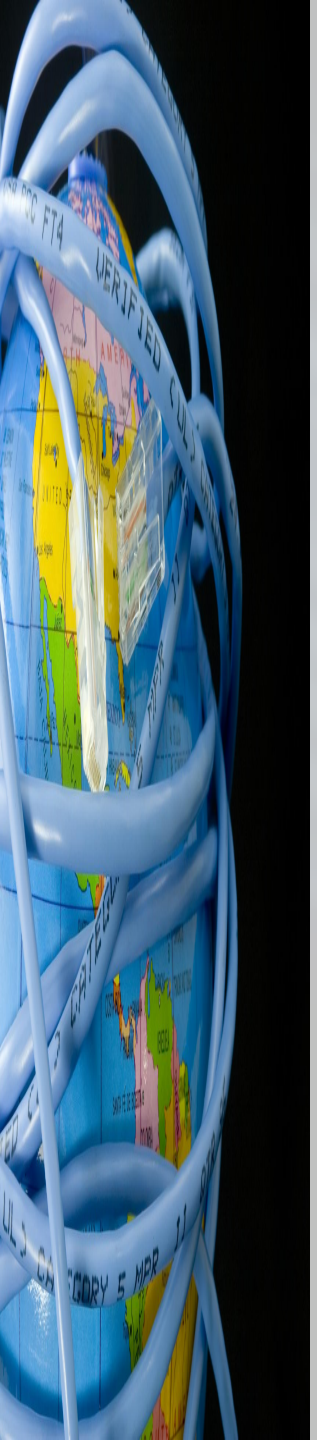
Пример оператора WHILE

```
i=0;
while(i<n)
{
      text+=" ";
       i++;
 }
```



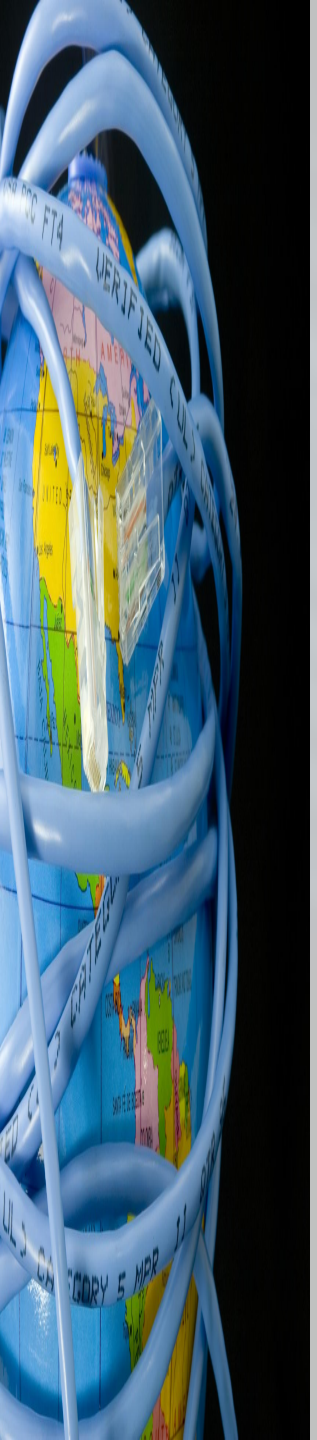
Прочие операторы

.	Доступ к полю объекта. (document.write(Buf);)
[ ]	Индексирование массива (dim[i])
( )	Изменение порядка вычислений или передача параметров функции
,	Разделение выражений в многократном вычислении



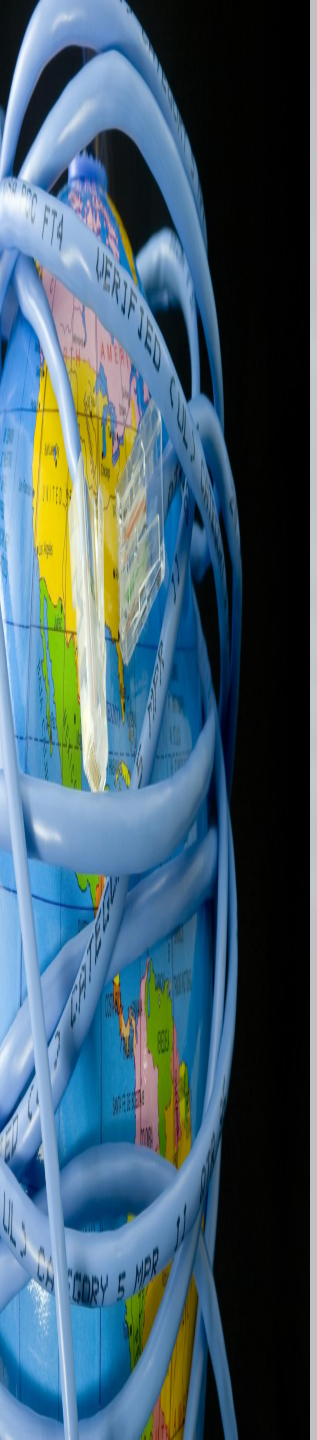
Объекты JavaScript

- Язык JavaScript является объектно-ориентированным.
- Объекты JavaScript представляют собой наборы свойств и методов. Можно сказать, что свойства объектов - это данные, связанные с объектом, а методы - функции для обработки данных объекта.
- В языке JavaScript имеется три вида объектов: встроенные объекты, объекты браузера и объекты, создаваемые программистом.



Встроенные объекты

- Array,
- Boolean,
- Date,
- Global,
- Function,
- Math,
- Number,
- String.

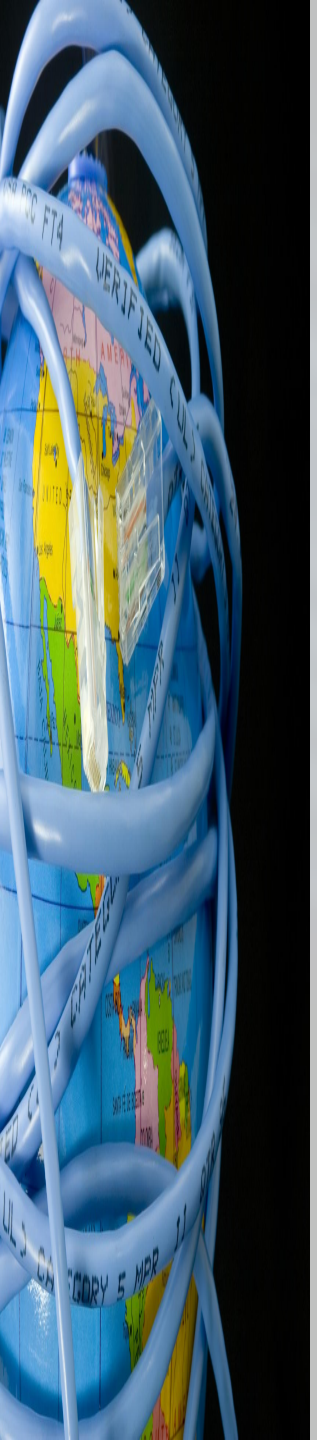


Встроенный объект Array. Массивы в JavaScript

- Массив в JavaScript является экземпляром встроенного объекта **Array**. Нумерация элементов в массиве начинается с нуля.
- Создать массив можно тремя способами:

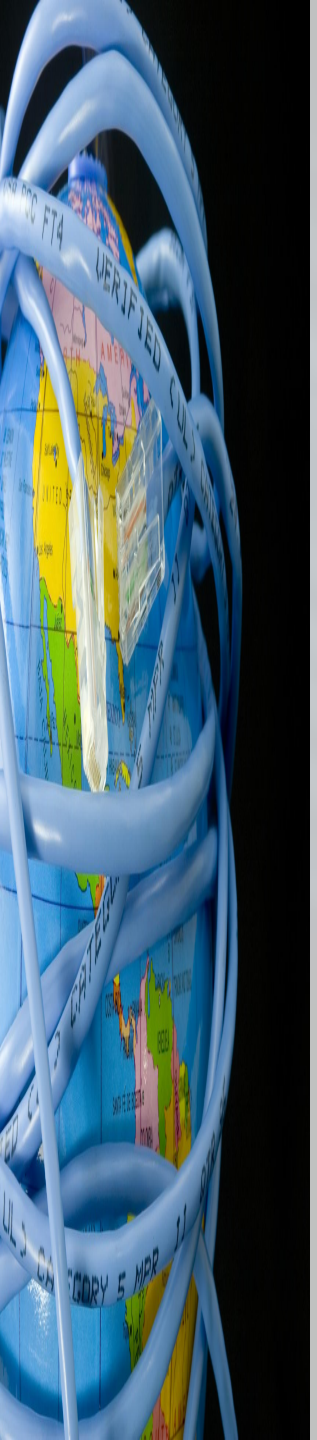
```
var a1 = new Array();
var a2 = new Array(3);
var a3 = new Array('раз','два','три');
```

- **a1** - массив, в котором нет ни одного элемента.
  a2 - массив из трех элементов с неопределенным (undefined) значением.
  a3 - массив, заданный списком своих элементов.



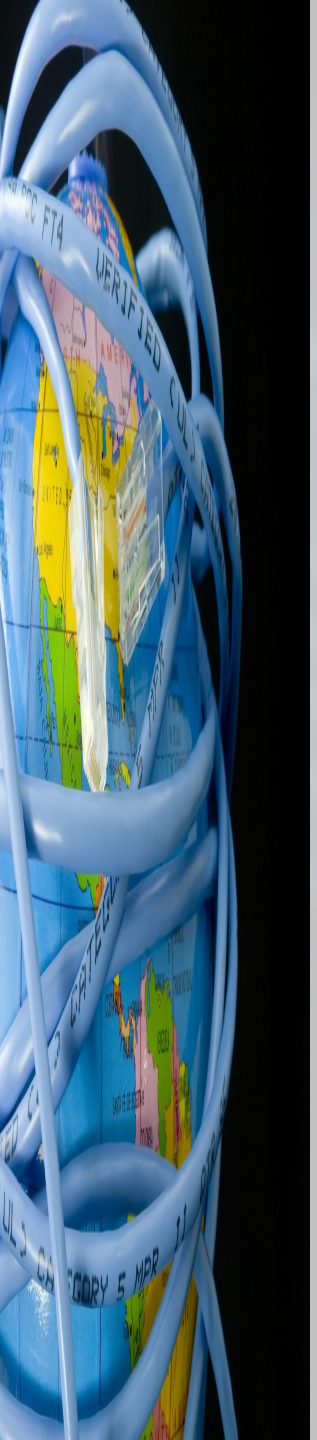
Свойство и методы объекта Array

- **length**. Число элементов массива.
- **concat( )**. Слияние двух массивов. Через параметр передается имя второго массива: c=a.concat(b); Здесь элементы массива b добавляются к элементам массива a.
- **join( )**. Слияние элементов массива в строку. Через параметр передается разделитель элементов. По умолчанию разделителем служит запятая. s=c.join('; ');
- **reverse( )**. Меняет порядок элементов массива на обратный.
- **slice( )**. Выделяет часть из массива. В качестве параметров передаются значения начального и конечного индексов, между которыми происходит выделение. При этом элемент массива с конечным индексом в результат не войдет. Следует помнить, что индексы отсчитываются от нуля.



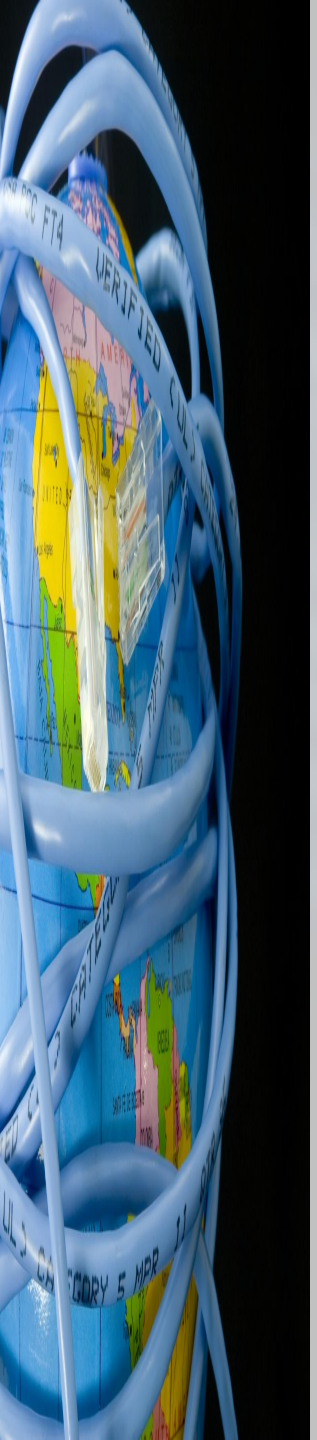
Объекты браузера

- Объекты браузера являются тем интерфейсом, с помощью которого сценарий JavaScript взаимодействует с посетителем и HTML-документом, загруженным в окно браузера, а также с самим браузером.
- Обращаясь к свойствам и методам этих объектов, можно выполнять различные операции над окном браузера, загруженным в это окно HTML-документом, а также над отдельными объектами, размещенными в HTML-документе.
- Практически в любом сценарии JavaScript необходимы такие объекты, как окно - **window** и документ -**document**.



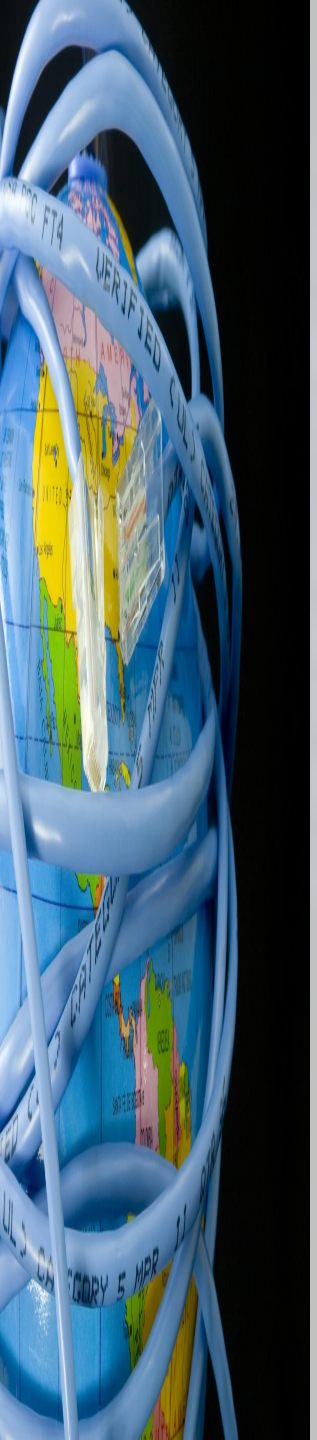
Свойства объекта window

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



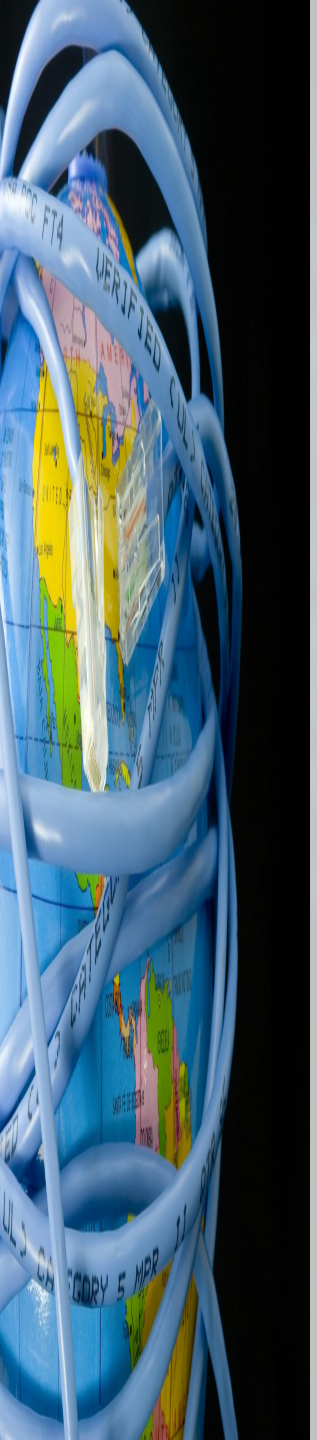
Методы объекта window

- **alert**. Отображение диалоговой панели **Alert** с сообщением и кнопкой **OK**. Через параметр передается сообщение, отображаемое в диалоговой панели. После вызова этого метода выполнение сценария задерживается до тех пор, пока посетитель не нажмет кнопку OK, расположенную в диалоговой панели.
- **confirm**. Отображение диалоговой панели Confirm с кнопками **OK** и **Отмена**. В зависимости от того, какая кнопка будет нажата, метод возвращает соответственно значение true или false.
- **prompt**. Отображение диалоговой панели **Prompt** с полем ввода и кнопками **OK** и **Отмена**. В зависимости от того, какая кнопка будет нажата, метод возвращает соответственно введенную строку или значение null. Метод имеет два параметра. Первый - сообщение над полем ввода. Второй (необязательный) - начальное значение строки ввода.



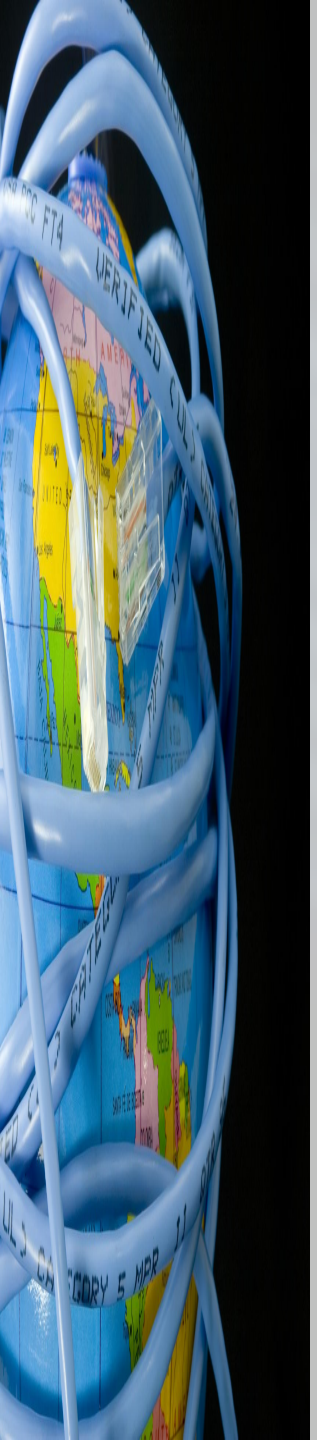
Методы объекта window

- **open**. Открытие окна. Метод имеет три параметра. Первый задает URL HTML-документа, предназначенного для загрузки в новое окно. Второй определяет имя окна для использования в атрибутеTARGET тега <A> или в атрибуте NAME тега <FORM>. Третий (необязательный) задает в виде текстовой строки параметры, определяющие внешний вид открываемого окна.
- **close**. Закрытие созданного или основного окна:
  newWindow.close();
  Текущее окно браузера можно закрыть одним из следующих способов:
  window.close(); self.close();
- **setTimeout**. Установка таймера. Применяется для ограничения времени ввода пароля, создания бегущих строк и всевозможных анимационных эффектов. Метод имеет два параметра. Первый задает выражение JavaScript, которое запускается по прошествии времени, указанного вторым параметром в миллисекундах. Заданное выражение



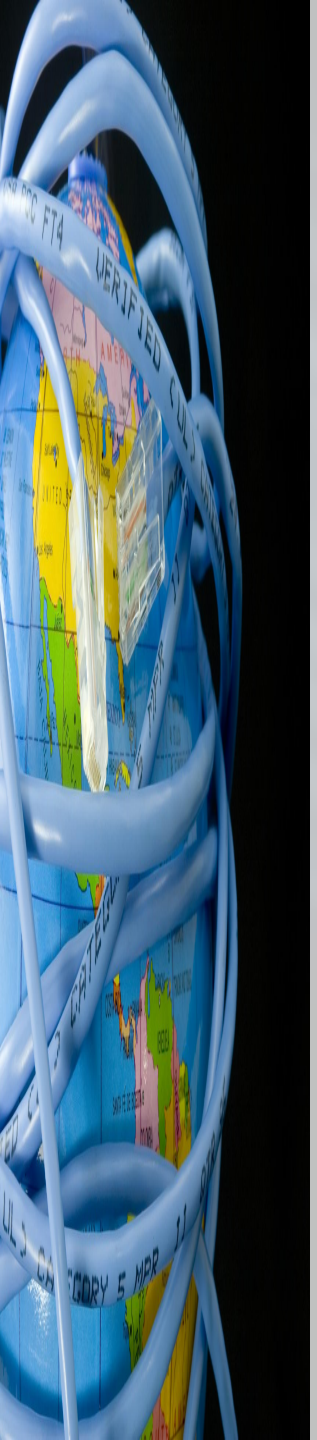
Методы объекта window

- **clearTimeout**. Сброс таймера. Для останова таймера
  метод **setTimeout** нужно вызвать с возвратом идентификатора, т.е.
  idTimer=setTimeout("change()", 2000);
  а затем этот идентификатор передать методу **clearTimeout** в качестве параметра:
  clearTimeout(idTimer);
- **blur( )**. При вызове метода окно теряет фокус.
- **focus( )**. При вызове метода окно получает фокус.



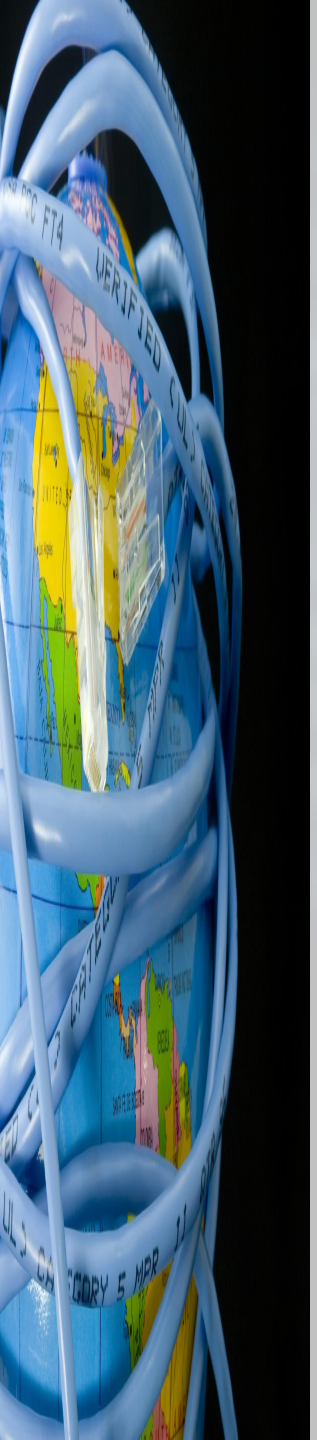
Методы объекта window

- **MoveTo(x,y)**. Перемещает окно в точку с координатами.
- **MoveBy(x,y)**. Перемещает окно на x пикселей по горизонтали вправо и на y пикселей вниз.
- **ResizeTo(x,y)**. Изменяет размер окна на указанные.
- **ResizeBy(x,y)**. Увеличивает или уменьшает размер окна на заданное количество пикселей.
- **print( )**. Печать документа.(не работает в IE 4)
- **scroll(x,y), ScrollTo(x,y)**. Прокручивает окно так, что точка с кординатами x,y становится левой верхней точкой окна.
- **ScrollBy(x,y)**. Прокручивает окно на x,y пикселей.
- **stop( )**. Прекращает загрузку документа в окно браузера.



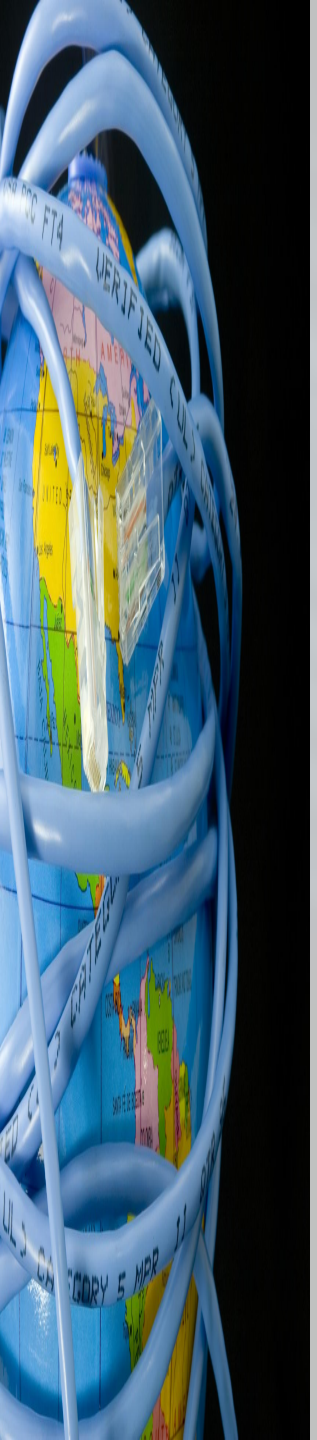
Свойства объекта document

- **URL**. Полный URL документа.
- **location**. Полный URL документа.
- **referrer**. URL вызывающего документа.
- **title**. Заголовок документа, определенный тегом <TITLE>.
- **bgColor**. Цвет фона документа.
- **fgColor**. Цвет текста.
- **linkColor**. Цвет ссылок.
- **alinkColor**. Цвет выбранных ссылок.
- **vlinkColor**. Цвет посещенных ссылок.



Свойства объекта document

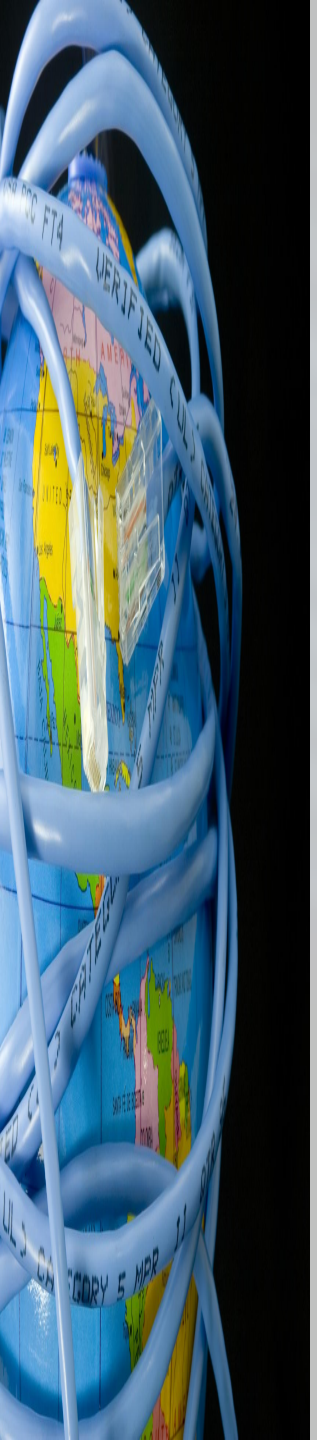
- **links**. Массив всех ссылок в документе.
- **anchors**. Массив локальных меток. Применяется для организации ссылок внутри документа.
- **applets**. Массив аплетов Java.
- **forms**. Массив форм в виде объектов.
- **images**. Массив растровых изображений.
- **embeds**. Массив объектов plug-in.
- **lastModified**. Дата последнего изменения документа.
- **cookie**. Значение cookie для текущего документа.



Свойства объекта document

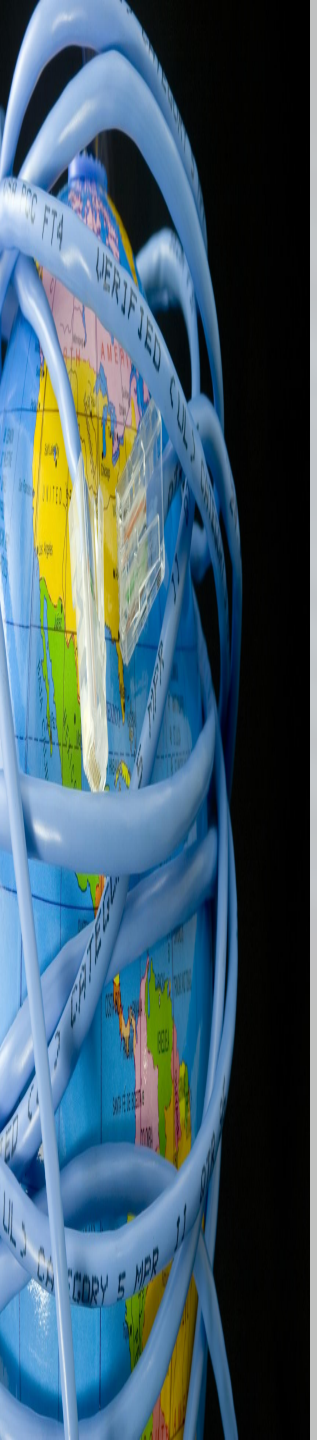
Объект document может содержать в себе другие объекты, доступные как свойства:

- **anchor**. Локальная метка, определенная тегом <A>.
- **form**. Форма, определенная тегом <FORM>.
- **history**. Список посещенных URL.
- **link**. Текст или изображение, играющие роль гипертекстовой ссылки, созданной тегом <A>, в котором дополнительно заданы обработчики событий onClick и onMouseOver.



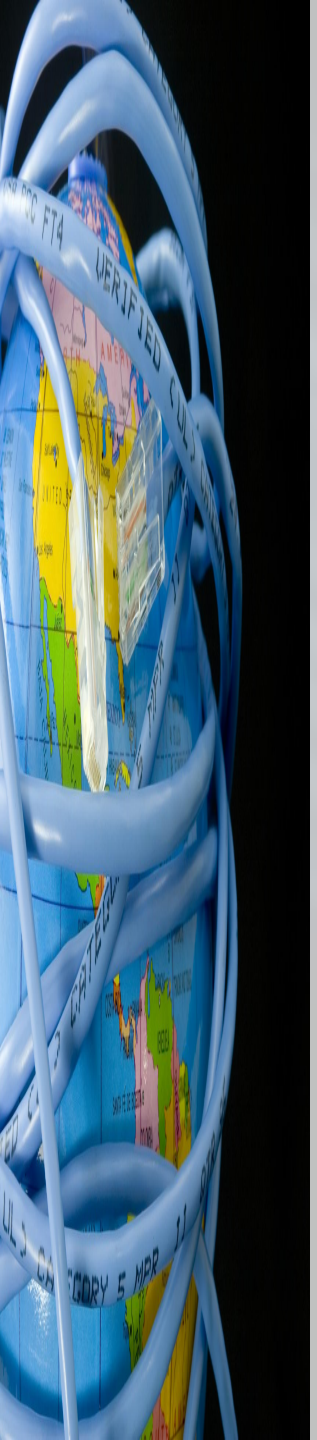
Методы объекта document

- **clear**. Удаление содержимого документа из окна просмотра.
- **write**. Запись в документ произвольной HTML-конструкции.
- **writeln**. Аналогичен **write**, но с добавлением символа перевода строки в конец строки.
- **open**. Открытие выходного потока для записи в HTML-документ данных типа MIME при помощи методов **write** и **writeln**.
- **close**. Закрытие потока данных, открытого методом **open**. В окне будут отображены все изменения содержимого документа, сделанные сценарием после открытия потока.



Ссылки в документе

- Для каждой ссылки, размещенной в HTML-документе, создается отдельный объект. Все такие объекты находятся в объекте document как элементы массива links. Анализируя эти элементы, сценарий JavaScript может определить свойства каждой ссылки в HTML-документе



Свойства ссылок в документе

- **length**. Количество ссылок в HTML-документе, т.е. количество элементов в массиве links.
- **hash**. Имя локальной ссылки, если она определена в URL.
- **host**. Имя узла и порт, указанные в URL.
- **hostname**. Имя узла и доменное имя узла сети. Если доменное имя недоступно, вместо него указывается адрес IP.
- **href**. Полный URL.
- **pathname**. Путь к объекту, указанный в URL.
- **port**. Номер порта, используемого для передачи данных с сервером, указанным в ссылке.
- **protocol**. Строка названия протокола передачи данных (включающая символ "двоеточие"), указанного в ссылке.
- **search**. Строка запроса, указанная в URL после символа "?".
- **target**. Имя окна, куда будет загружен документ при выполнении ссылки. Это может быть имя существующего окна фрейма, определенного тегом <FRAMESET>, или одно из зарезервированных