Основы JavaScript

JavaScript

JS

- Скриптовый
- Объектно-ориентированный
- Динамически типизированный
- Функции – объекты первого класса
- «Сборка мусора»
- Является реализацией стандарта ECMAScript

http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

Из истории JavaScript

- Когда в 1995 г. создавался язык JavaScript, у него изначально было другое название: «LiveScript». Первоначально язык вообще назывался Mocha. Но тогда был очень популярен язык Java, и маркетологи решили, что схожее название сделает новый язык более популярным
- *JavaScript* изначально создавался для того, чтобы сделать web-странички «живыми».

Назначение JavaScript

- **Интерактивность**
- Изменение документа после его загрузки
- Проверка вводимой пользователем информации
- Отображение диалоговых окон и сообщений

JavaScript

- JavaScript может выполняться не только в браузере, а где угодно, нужна лишь специальная программа – интерпретатор. Процесс выполнения скрипта называют «интерпретацией».

JavaScript

- Современные интерпретаторы перед выполнением преобразуют JavaScript в машинный код или близко к нему, оптимизируют, а уже затем выполняют. И даже во время выполнения стараются оптимизировать. Поэтому JavaScript работает очень быстро.

ОСНОВЫ JAVASCRIPT

Основы JavaScript

Для вставки скриптов на JavaScript используется тег <script>

1. <script language="JavaScipt">
   … // код
   </script>
2. <script src="script.js"></script>

Основы JavaScript

```
<html>
<head><title>Заголовок</title>
<script language=JavaScipt>
      //функции
</script>
</head>

<body>
<script>
      //вызов функций
</script>
</body>
</html>
```

Основы JavaScript

- JavaScript – C-подобный язык
- JavaScript – регистрозависимый язык (в отличие от HTML)
- JavaScript – объектно-ориентированный язык

Понятия

JS

- Объект – совокупность свойств, методов, коллекций и событий, предоставляемых браузером в рамках объектной модели
- Свойство – переменная в рамках объекта, используемая для получения значения или установки нового
- Метод – функция, предоставляемая объектом для выполнения каких-либо действий
- Событие – какое-либо действие пользователя или момент работы браузера (для реакции на события создаются обработчики событий)

Типы данных

- Число «number»
- Строка «string»
- Логический тип «boolean»
  (значение true или false)
- Специальное значение «null»
  (значение неизвестно)
- Специальное значение «undefined»
  (значение не присвоено)
- Объект «object» (в том числе массив)

Пять примитивных типов + объекты

Число «number»

JS

- Все числа в JavaScript с плавающей точкой.
- 8 байт (формат по стандарту IEEE 754.1)

1. Целые литералы

```
7
0
100500
```

2. Вещественные литералы

```
3.14
.33333333333333333
6.02e23  //6.02×1023
1.4738223E-32  //1.4738223×10-32
```

Число «number»

JS

- Проблема
- `( 0.1 + 0.2 == 0.3 ); // false`
- `( 0.1 + 0.2 ); // 0.30000000000000004`

- Решение
- `(0.1 * 10 + 0.2 * 10) / 10 ; // 0.3`
  `(0.1 + 0.2).toFixed(10); // 0.3, округление до 10 знаков`

Строка «string»

JS

- Последовательность 16-битных значений, обычно символов UNICODE
- Нумерация символов с 0
- Строки - символы в одинарных или двойных кавычках
- Строковые литералы

```
‘Hello\nWorld’  // в две строки
“100500”
```

Переменные

JS

```
var message; // объявим переменную
message = 'Hello'; // сохраним в переменной строку
```

Новый синтаксис ES2015:

```
let message; // объявим переменную с блочной видимостью
```

- *Переменная* состоит из имени и выделенной области памяти, которая ему соответствует.

Объекты

JS

```
var o = {}; // пустые фигурные скобки
o.width = 300;
o.height = 200;
o.name = ‘Виктор Петрович’;
```

- *Ассоциативный массив*: структура, пригодная для хранения любых данных
- **В JavaScript всё объекты**

Объекты

```
var user = {
    name: "Таня",
    age: 25,
    size: {
        top: 90,
        middle: 60,
        bottom: 90
    }
}
```

- Доступ через квадратные скобки

alert(**user['name']**) // "Таня"

- Доступ к свойству через переменную

alert(**user.name**) // "Таня"

alert(**user.size.top**) // “90"

Массивы

JS

```
var arr = []; // пустые квадратные скобки
arr = ["Яблоко", "Апельсин", "Слива"];
// сохраним в массив 3 строки
alert(arr[1]); // выведет "Апельсин"
```

- *Структура данных* которая предназначена для хранения коллекций (пронумерованных значений)

Массивы

JS

- Элемент можно заменить или добавить

```
arr[2] = 'Груша';
arr[3] = 'Лимон';
// ["Яблоко", "Апельсин", "Груша", "Лимон"]
```

ОСНОВНЫЕ УПРАВЛЯЮЩИЕ СТРУКТУРЫ

Операторы сравнения

JS

- Больше/меньше: a > b, a < b.
- Больше/меньше или равно: a >= b, a <= b.
- Равно a == b. Для сравнения используется два символа равенства '='. Один символ a = b означал бы присваивание.
- «Не равно». В математике он пишется как ≠, в JavaScript – знак равенства с восклицательным знаком перед ним !=.

https://learn.javascript.ru/comparison

Оператор if

JS

```
if (year == 2017) {
  alert( 'Да, сейчас 2017' );
} else {
  alert( ‘Нет, сейчас не 2017' );
}
```

- Число 0, пустая строка "", null и undefined, а также NaN являются false,
- Остальные значения – true.

Оператор switch

JS

```
switch (year) {
  case 2017: alert( 'Сейчас 2017' );
  break;
  case 2018: alert( 'Сейчас 2018' );
  break;
  default: alert( 'Не в курсе о таком' );
  break;
}
```

Оператор for

```
for (начало; условие; шаг) {
    // ... тело цикла ...
}
```

```
var i;
```

Пример:

```
for (i = 0; i < 7; i++) {
    alert( i );
}
```

Прерывание цикла: break

Следующая итерация: continue

Обработка ошибок

JS

```
try {
  // код с потенциальными ошибками
}
catch (err) {
  // обработка ошибки
  // выполняется в случае ошибке в try
}
finally {
  // выполняется в любом случае
}
```

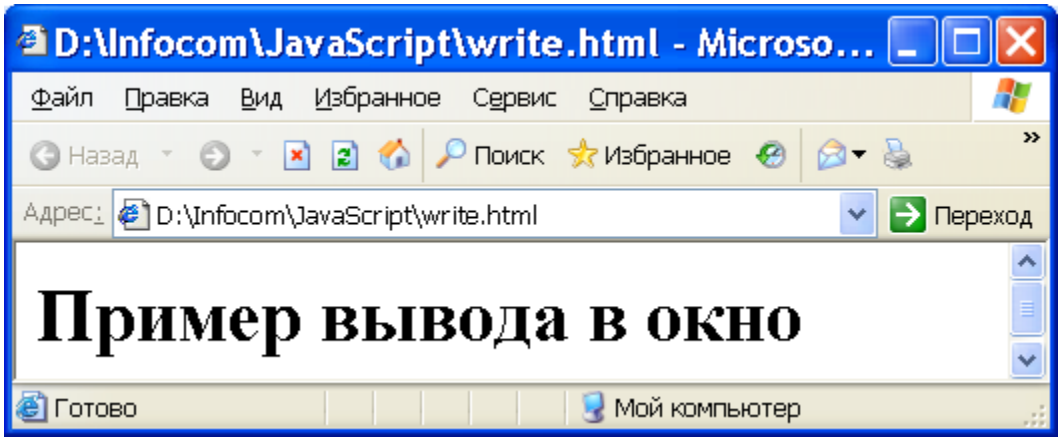
Ввод/вывод

ВЗАИМОДЕЙСТВИЕ С ПОЛЬЗОВАТЕЛЕМ

Метод write()

- Средства вывода - выводит параметры метода в окно браузера

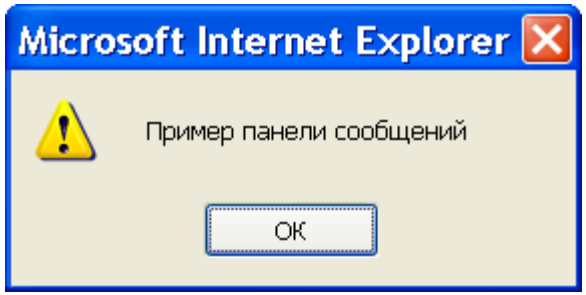
```
<script>
    window.document.write("<h1>Hello World</h1>")
</script>
```



Метод alert()

- Средства вывода

```
<script>
    window.alert("Пример сообщения");
</script>
```



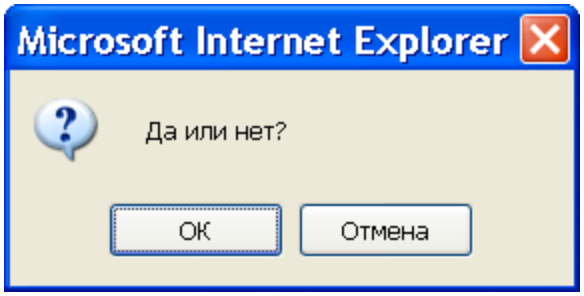
Метод confirm()

JS

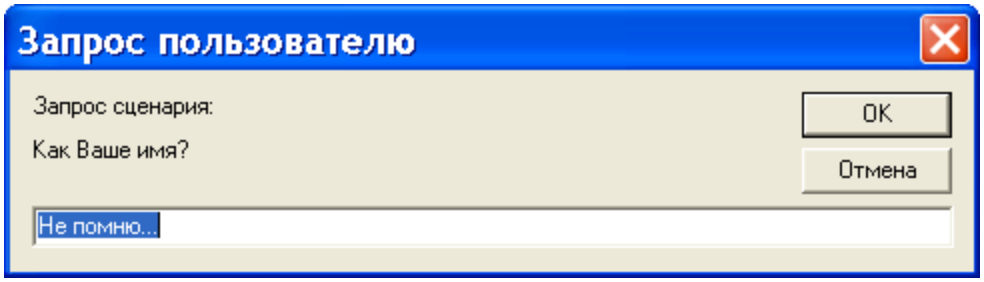
- Средства ввода

```
<script>
res = window.confirm("Да или нет?");
console.log("res = " + res);
</script>
```

- Метод возвращает значения true (1) или false (0)



Метод prompt()



- Средства ввода

```
<script>
    res = window.prompt("Как Ваше имя?",
        "Не помню...");
    console.log("res="+res);
</script>
```

- Метод возвращает содержимое строки ввода или null

Глобальные встроенные объекты

ОБЪЕКТЫ В JAVASCRIPT

Объекты в JS


Виды объектов
Встроенные объекты
Объекты, составляющие объектную модель браузера
Пользовательские объекты

Встроенные объекты

- Объект Array
  предназначен для работы с массивами
- Объект Date
  предназначен для работы с датой и временем
- Объект String
  предназначен для работы со строками
- Объект Math
  предназначен для выполнения математических функций
- Объект RegExp
  предназначен для работы с регулярными выражениями
- Объект Console
  предназначен для отладки программного кода

Объекты браузера

JS

- Объект window
  предназначен для работы с окном браузера
- Объект document
  предназначен для работы с html документов
- Объект history
  предназначен для работы с историей

Объект Date

JS

- Дата представляет число миллисекунд, прошедших с 1 января 1970 года
- 1 сутки = 86 400 000 миллисекунд
- У объекта Date свойств нет
  - today= new Date()
  - today= new Date("Jan 1 2011 00:00:01")
  - today= new Date(2011, 11, 31)

Объект Date

JS

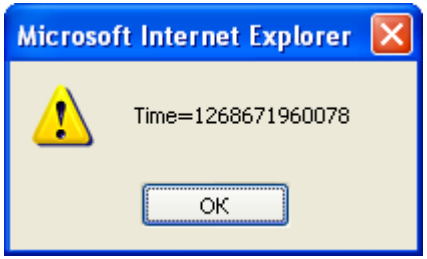
- Методы даты и времени
  - setDay() день недели (0..6) вск..суб
  - setDate() день (1..31)
  - setMonth() месяц (0..11)
  - setTime() время в миллисекундах

  - getDay() день недели
  - getDate() день
  - getMonth() месяц
  - getTime() время в миллисекундах

Объект Date

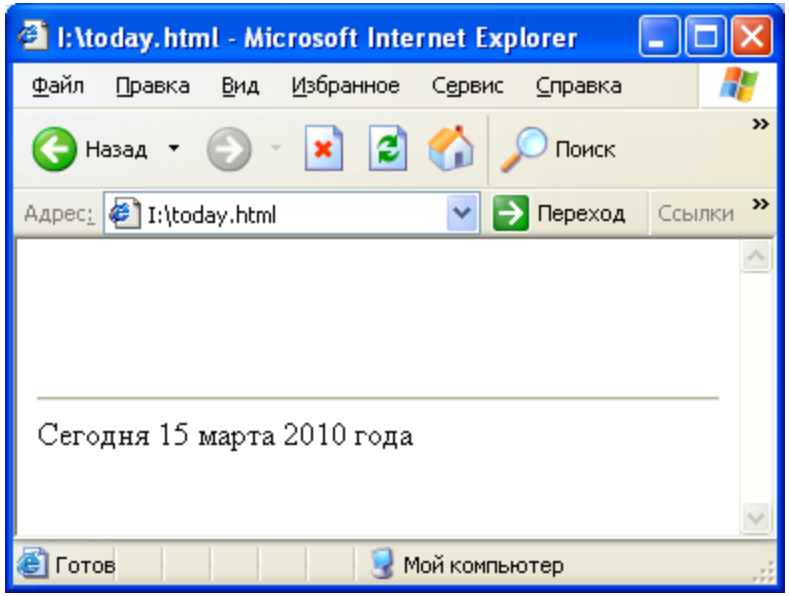
JS

- today=new Date()
  alert ("Day="+today.getDay())
  alert ("Time="+ today.getTime())



Текущая дата

```
Months = new Array("января","февраля",...,"декабря")
today = new Date()
document.write("<hr>")
document.write("Сегодня "+
today.getDate()+
" "+Months[today.getMonth()]+
" "+today.getYear()+" года")
```



Объект String

JS

- Свойство
  - length
- Методы
  - charAt() – возвращает символ, находящийся в указанной позиции
  - indexOf() – возвращает позицию подстроки в строке,
    -1, если подстрока не найдена

  и множество других методов

Объект Math

```
Math.pow(2,53)  //=>9007199254740992:  2 в степени 53
Math.round(.6)  //=>1.0:округление до ближайшего целого
Math.ceil(.6)   //=>1.0:округление вверх
Math.floor(.6)  //=>0.0:округление вниз
Math.abs(-5)    //=>5:абсолютное значение
Math.max(x,y,z) //Возвращает наибольший аргумент
Math.random()   //Псевдослучайное числох,где 0<=x<1.0
Math.sqrt(3)    //Корень квадратный из 3
Math.pow(3,1/3) //Корень кубический из 3
```

ФУНКЦИИ

Функции

JS

```
// Function Declaration <- Предпочтительно
function sum(a, b) {
    return a + b;
}

// Function Expression
var sum = function(a, b) {
    return a + b;
}
```

Функции

```
function showMessage() {
    alert( 'Привет всем присутствующим!' );
}
showMessage();
showMessage();
```

Локальные переменные

```
function showMessage() {
  // локальная переменная message
  var message = 'Привет, я - Вася!';
  alert( message );
}
showMessage(); // 'Привет, я - Вася!'
alert( message ); // ошибка, переменная невидна
```

Внешние переменные

JS

```
var userName = 'Вася';
function showMessage() {
  var message = 'Привет, я ' + userName;
  alert(message);
}

showMessage(); // Привет, я Вася
```

Параметры

JS

```
function showMessage(from, text) {
    from = "** " + from + " **"; //  добавим звездочек
    alert(from + ': ' + text);
}
showMessage('Маша', 'Привет!');
showMessage('Маша', 'Как дела?');
```

Возврат значения

```
function checkAge(age) {
  if (age > 18) {
    return true;
  } else {
    return confirm('Родители разрешили?');
  }
}
var age = prompt('Ваш возраст?');
if (checkAge(age)) {
  alert( 'Доступ разрешен' );
} else {
  alert( 'В доступе отказано' );
}
```