Основы JavaScript

JavaScript

Основы JavaScript

1. Структура кода
2. Переменные и типы
3. Взаимодействие с посетителем
4. Особенности операторов
5. Логические операторы
6. Циклы
7. Конструкция switch
8. Функции
9. Методы и свойства

Структура кода

- Операторы разделяются точкой с запятой:

```
1  alert('Привет'); alert('Мир');
```

- Перевод строки тоже подразумевает точку с запятой:

```
1  alert('Привет');
2  alert('Мир');
```

- Поддерживаются однострочные комментарии // ... и многострочные /* ... */

Переменные и типы

- Объявляются директивой var.
- Могут хранить любое значение:
- var x = 5;
- x = "Петя";

Переменные и типы

- Есть 5 «примитивных» типов и объекты:
- 1 x = 1; // число
- 2 x = "Тест"; // строка, кавычки могут быть одинарные или двойные
- 3 x = true; // булево значение true/false
- 4 x = null; // **«**ссылкой на нулевой адрес/объект»
- 5 x = undefined; // **«переменная не присвоена»**

Переменные и типы

- Также есть специальные числовые значения Infinity (бесконечность) и NaN.
- Значение NaN обозначает ошибку и является результатом числовой операции, если она некорректна.

- В имени переменной могут быть использованы любые буквы или цифры, но цифра не может быть первой.

- Символы доллар $ и _ подчёркивание допускаются наравне с буквами.

Взаимодействие с посетителем

- Простейшие функции для взаимодействия с посетителем в браузере:
- prompt(вопрос[, по_умолчанию])
- Задать вопрос и возвратить введённую строку, либо null, если посетитель нажал «Отмена».
- confirm(вопрос)
- Задать вопрос и предложить кнопки «Ок», «Отмена». Возвращает, соответственно, true/false .
- alert(сообщение)
- Вывести сообщение на экран.

Взаимодействие с посетителем

- Все эти функции являются *модальными*, т.е. не позволяют посетителю взаимодействовать со страницей до ответа.

- Например:

```
1   var userName = prompt("Введите имя?", "Василий");
2   var smokes = confirm("Вы хотите чаю?");
3
4   alert( "Посетитель: " + userName);
5   alert( "Чай: " + smokes);
```

Операторы JavaScript

- **Для сложения строк используется оператор +.**Если хоть один аргумент — строка, то другой тоже приводится к строке:

```
1  alert( 1 + 2 );   // 3, число
2  alert( '1' + 2 ); // '12', строка
3  alert( 1 + '2' ); // '12', строка
```

Операторы JavaScript

- **Сравнение === проверяет точное равенство, включая одинаковый тип.**
- Это самый очевидный и надёжный способ сравнения.

- **Остальные сравнения**
- **== < <= > >= осуществляют числовое приведение типа:**
-

```
1    alert( 0 == false ); // true
2    alert( true > 0 );   // true
```

- Исключение — сравнение двух строк

Операторы JavaScript

Сравнение строк — лексикографическое, символы сравниваются по своим unicode-кодам.

- Поэтому получается, что строчные буквы всегда больше, чем прописные:
-
- 1 alert('a' > 'Я'); // true

Операторы JavaScript

- **Исключение:**
  значения null и undefined ведут себя в сравнениях не как ноль.
- Они равны null == undefined друг другу и не равны ничему ещё. В частности, не равны нулю.
- В других сравнениях (кроме ===)
  значение null преобразуется к нулю,
  а undefined — становится NaN («ошибка»).

Операторы JavaScript

- Такое поведение может привести к неочевидным результатам, поэтому лучше всего использовать для сравнения с ними ===. Оператор == тоже можно, если не хотите отличатьnull от undefined.
- Например, забавное следствие этих правил для null:
-
- 1alert(null > 0); // false, т.к. null преобразовано к 0
- 2alert(null >= 0); // true, т.к. null преобразовано к 0
- 3alert(null == 0); // false, в стандарте явно указано, что null равен лишь undefined
- С точки зрения здравого смысла такое не возможно. Значение null не равно нулю и не больше, но при этом null >= 0 возвращает true!

Логические операторы

JavaScript

В JavaScript есть логические операторы:
И (обозначается &&),
ИЛИ (обозначается ||) и
НЕ (обозначается !).

Они интерпретируют любое значение как логическое.

Результатом логического оператора служит последнее значение в коротком цикле вычислений.

Операторы Цикла JavaScript

- Поддерживаются три вида циклов:
- 01// 1
- 02while (условие) {
- 03 ...
- 04}
- 05
- 06// 2
- 07do {
- 08 ...
- 09} while(условие);
- 10
- 11// 3
- 12for (var i = 0; i < 10; i++) {
- 13 ...
- 14}

Операторы Цикла JavaScript

- Поддерживаются три вида циклов:
- Переменную можно объявлять прямо в цикле, но видна она будет и за его пределами.
- Поддерживаютсядирективы break/continue для выхода из цикла/перехода на следующую итерацию.

Операторы Цикла JavaScript

- Для выхода одновременно из нескольких уровней цикла можно задать метку.
- Синтаксис: «имя_метки:», ставится она только перед циклами и блоками, например:

```
1    outer:
2    for(;;) {
3      ...
4     for(;;) {
5       ...
6      break outer;
7          }
8     }
```

- Переход на метку возможен только изнутри цикла, и только на внешний блок по отношению к данному циклу. В произвольное место программы перейти нельзя.

Операторы JavaScript

Конструкция switch

- Конструкция switch заменяет собой сразу несколько if.
- Это — более наглядный способ сравнить выражение сразу с несколькими вариантами.

Операторы JavaScript

Конструкция switch

```
01 switch(x) {
02   case 'value1':  // if (x === 'value1')
03     ...
04     [break]
05
06   case 'value2':  // if (x === 'value2')
07     ...
08     [break]
09
10   default:
11     ...
12     [break]
13 }
```

- Переменная x проверяется на строгое равенство первому значению value1, затем второму value2 и так далее.
- Если соответствие установлено — switch начинает выполняться от соответствующей директивы caseи далее, *до ближайшего break* (или до конца switch).При этом case называют *вариантами switch*.
- Если ни один case не совпал — выполняется (если есть) вариант default.

Функции JavaScript

- Синтаксис функций в JavaScript:
- 1 // function имя(список параметров) { тело }
- 2 function sum(a, b) {
- 3 var result = a + b;
- 4
- 5 return result;
- 6 }
- sum — имя функции, ограничения на имя функции — те же, что и на имя переменной.
- Переменные, объявленные через var внутри функции, видны везде внутри этой функции, блоки if, for и т.п. на видимость не влияют.

Функции JavaScript

- Параметры передаются «по значению», т.е. копируются в локальные переменные a, b, за исключением объектов, которые передаются «по ссылке», их мы подробно обсудим в главе Объекты как ассоциативные массивы.
- Функция без return считается возвращающей undefined. Вызов return без значения также возвращает undefined:

```
1  function f() { }
2  alert( f() ); // undefined
```

Методы и свойства

JavaScript

- Все значения в JavaScript, за исключением null и undefined, содержат набор вспомогательных функций и значений, доступных «через точку».

- Такие функции называют «методами», а значения — «свойствами».

Методы и свойства

JavaScript

- Например:
-
- ```
  1    alert( "Привет, мир!".length ); // 12
  ```
- Еще у строк есть *метод* toUpperCase(), который возвращает строку в верхнем регистре:
-
- ```
  1    var hello = "Привет, мир!";
  2
  3    alert( hello.toUpperCase() ); //"ПРИВЕТ, МИР!"
  ```