JavaScript: переменные и операции



ORTDNIPRO.ORG/WEB

1. Наши планы на JavaScript

Наши планы на JavaScript

Переменные и операции
Ветвления (условные операторы)
Циклы / Массивы (структуры данных)
Функции
Объекты

Основы программирования



Управление документом (DOM) Событийная модель в JavaScript Разработка интерактивных виджетов.

Прикладное применение

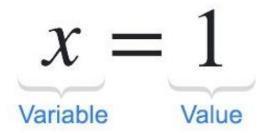
2. Переменные

Задача любой программы – обработка данных

```
let price = 799;
let quantity = 10;
let totalCost = price * quantity;
console.log('Total Cost:', totalCost, 'UAH.');
let totalCost = price * quantity;
```

Перед тем как использовать, **переменную** нужно **объявить**. Сказать браузеру, что мы хотим создать еще одну «*коробочку*» для значений и дать ей имя. Объявляются переменные при помощи ключевого слова **let**.

Для хранения данных (информации), в JavaScript используются переменные. Переменные можно представить как «коробочку» у которой есть название и в которой хранится какоенибудь значение. Значением может быть число, строка или другие типы данных поддерживаемые JavaScript.



3. Ввод/вывод данных

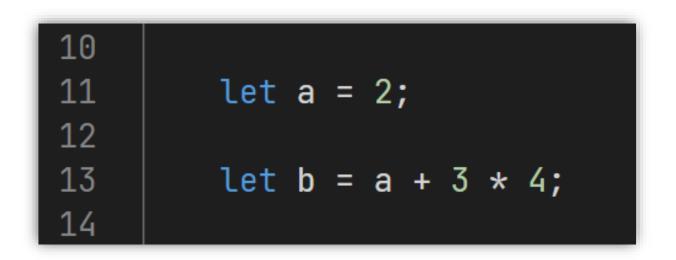
Ввод/вывод данных

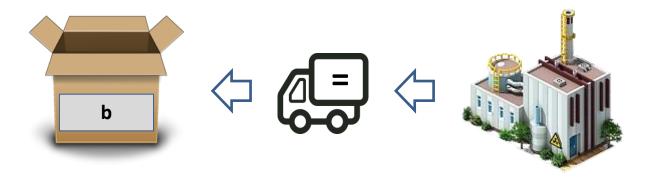
```
//Варианты ввода данных (без разметки)
       let userName = prompt('Please enter user name');
       let canDrive = confirm('are you can drive?');
 5
 6
       //Варианты вывода данных (без разметки)
 8
       alert('Hello');
 9
       console.log('User name:', userName);
10
11
       document.write(`<h1>Hello ${userName}</h1>`);
12
13
14
```

Основной способ ввода данных в JavaScript приложения – формы и элементы ввода, а вывод – в разметку документа. В тоже время в учебных (и **отладочных**) целях нам будут полезны следующие способы ввода/вывода данных (информации).

4. Операции, операторы, операнды и выражения

Оператор присваивания





Чтобы указать, что именно нужно записать в перемененную используется **оператор присваивания**.

Оператор присвоения берёт то, что справа от него и записывает в переменную имя которой расположено слева от него. Если справа расположено выражение, то первым делом оно будет рассчитано, и в переменную попадёт уже результат расчёта выражения.

Выражения

```
2
       let a = 3;
 3
 4
       let b = a + 6;
 5
 6
       let c = b + 1;
 8
       c = a + b * c + 7;
 9
       console.log('In Variable C:', c);
10
11
12
```

По правую сторону от оператора присвоения может быть как конкретное значение, а также может быть выражение – формула рассчитав которую компьютер получит результат который будет записан в переменную имя которой стоит слева от знака присвоения. В выражении могут участвовать как и конкретные значения (константы) так и другие переменные.

Операторы, операнды и операции...

Для выполнения действий (**операций**) над переменными (или значениями) используются **операторы** (которых) существует довольно много). С некоторыми из них все знакомы, например с арифметические операторами.

У операторов есть приоритеты, какой приоритет выше, какой ниже запомнить непросто. Поэтому в случае сомнений какая операция будет первой а какая второй — смело используйте скобки. Принцип их применения такой же как и в математике — скобки повышают приоритет операции в них записанной.

«Скобками программу не испортишь» (с)

Операторы и их приоритеты

Level	Operators
1	0 🛮 -
2	1 ~ - ++
3	* / %
2 3 4 5 6 7 8	+ -
5	<< >> >>>
6	< <= > >=
7	== !=
8	&
9	٨
10	1
11	&&
12	II
13	?:
14	= += -= *= /= %= <<=
	>>= >>>= &= ^= =

У операторов есть приоритеты, какой приоритет выше, какой ниже запомнить непросто. Поэтому в случае сомнений какая операция будет первой а какая второй – смело используйте скобки. Принцип их применения такой же как и в математике – скобки повышают приоритет операции в них записанной.

5. Типы данных (string & number)

Типы данных (переменных)

```
1
2    let a = 7;
3
4    let b = 8;
5
6    let c = a + b;
7
8    let d = a * b;
9
10    console.log('Value in C:', c); //15
11
12    console.log('Value in D:', d); //56
13
```

```
1
2    let a = '7';
3
4    let b = '8';
5
6    let c = a + b;
7
8    let d = a * b;
9
10    console.log('Value in C:', c); //'78'
11
12    console.log('Value in D:', d); //56
13
```

```
let a = '7';

let b = 8;

let c = prompt('Enter Some Number');

console.log(typeof(a), typeof(b), typeof(c)); //string, number, string
```

В JavaScript отсутствует жёсткая типизация данных, при которой тип переменной определяется при её объявлении. В JavaScript тип переменной определяется при присваивании ей значения. И может меняться при каждом новом присвоении. Мы можем узнать тип переменной воспользовавшись функцией **typeof(...)**.

Типы данных (переменных)

Тип данных – пометка для компьютера как относиться к тем или иным данным и какие операции с ними возможно проводить.

Тип определяет **возможные значения** и их «**смысл**», а также **операции** которое возможно выполнять над этими значениями.

undefined, number, string, boolean, function, object, symbol, bigint

6. Преобразование типов

Преобразование типов

Функции parseInt()/parseFloat() позволяют преобразовать тип переменной со строкового на числовой (это возможно если в строке действительно содержатся хоть какиенибудь цифровые символы, иначе результатом будет значение NaN). parseInt() — работает с целыми числами, parseFloat() поддерживает дробные числа.

Также для преобразования типов может быть использован оператор + в унарном виде, но в отличии от parse*-функций любые нецифровые символы в строке приведут к получению значения NaN.

7. NaN (Not a Number)

NaN – Not a Number

```
let a = 'hello';
 5
       let b = 7;
 6
 8
       let c = a * b; //NaN - в результате
       выполнения арефетической операции среди
       операндов оказалось значение, которое
       невозможно преобразовать к числу;
 9
       let d = parseFloat(a); //NaN - невозможно
10
       даже часть строки преобразовать к числу;
```

NaN (Not a Number) – специальное значение типа number которое показывает, что в результате выполнения арифметической операции (или явного преобразования к числу) один из операндов не удалось успешно преобразовать к числу. Поскольку JavaScript не типизированный язык то ошибок преобразования типов в нём быть не может, потому и существует такая конструкция как **NaN**.

8. Math

Объект Math

```
let a = 7.6374;
 5
 6
       let b = Math.round(a);
 8
       let c = Math.floor(a);
 9
10
       let d = Math.ceil(a);
11
       console.log(a, b, c, d);
12
13
       // 7.6374 8 7 8
14
```

Объект **Math** содержит большое количество математических функций, в частности функции округления...

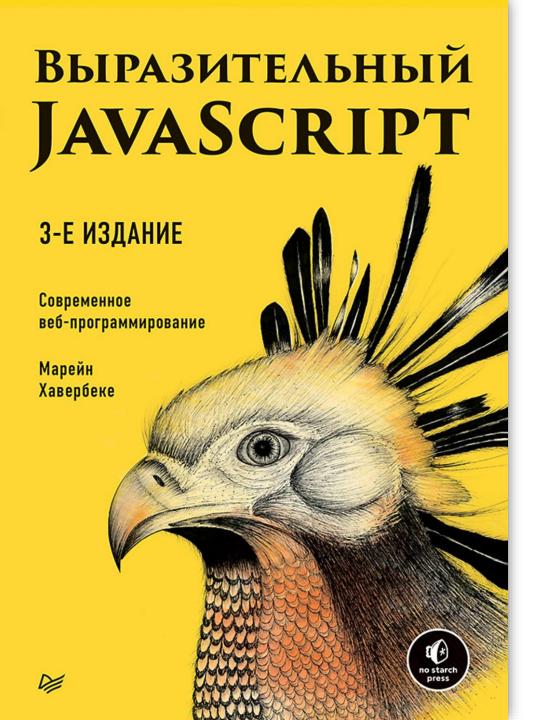
9. Немного практики #1

Задача: Разработать скрипт который на основании дохода физ. лица рассчитает суммы налогов которые ему необходимо заплатить и сколько у него останется после уплаты налогов.

10. Немного практики #2

Задача: Разработать скрипт который рассчитывает **индекс массы тела** пользователя.

Будет полезным



Выразительный JavaScript.

Современное веб-программирование

Марейн Хавербеке

Замечательная книга, как для введения в программирование, так и освоения JavaScript.

Домашнее задание Тренируемся!

Программирование ремесло и требует тренировки...

- 1) Задаётся сторона квадрата. Найти его периметр;
- 2) Задаётся длина ребра куба. Найти объем куба и площадь его боковой поверхности;
- 3) Задаётся радиус окружности. Найти длину окружности и площадь круга;
- 4) Задаются объем и масса вещества. Определить плотность материала этого вещества;
- 5) Известны количество жителей в государстве и площадь его территории (в км2). Определить плотность населения в этом государстве.
- 6). Даны катеты прямоугольного треугольника. Найти его гипотенузу.
- 7) Рассчитать значение **у**, при любых введённых значениях **a**: $y = \frac{a^2 + 1}{\sqrt{a^2 + 1}}$



На следующем занятии

JS: условия и ветвление

